Instituto Tecnológico de Costa Rica

Escuela de Ingeniería Electrónica

Diseño e implementación de control a través de matrices de cinemática para el brazo robótico SCORBOT ER 4U ubicado en el Instituto Tecnológico de Costa Rica

Informe de Proyecto de Graduación para optar por el título de Ingeniero en Electrónica con el grado académico de Licenciatura

Ruth Iveth Campos Artavia

San Carlos, Junio de 2019

INSTITUTO TECNOLOGICO DE COSTA RICA
ESCUELA DE INGENIERIA ELECTRONICA
PROYECTO DE GRADUACIÓN
TRIBUNAL EVALUADOR
ACTA DE EVALUACIÓN

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

Estudiante: CAMPOS ARTAVIA RUTH

Carné: 2013026084

Nombre del Proyecto: *"Diseño e implementación de control a través de matrices de cinemática para el brazo robótico SCORBOT ER 4U ubicado en el Instituto Tecnológico de Costa Rica"*

Miembros del Tribunal

Ing. Luis Diego Gómez Rodríguez

Profesor lector

Ing. Pablo César Rodríguez Vargas

Profesor lector

Ing. Luis Miguel Esquivel Sancho

Profesor asesor

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

Nota Final del Proyecto de Graduación : 90

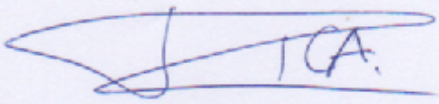
San Carlos, 18 de junio 2019

Declaratoria de Autenticidad

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

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

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



Ruth Iveth Campos Artavia
Céd: 207390609

San Carlos, 18 de Junio 2019

Resumen

En este documento pretende realizar la descripción de la actualización tecnológica llevada a cabo en el Instituto Tecnológico de Costa Rica en el Campus Tecnológico Local San Carlos al un robot manipulador SCORBOT ER-4U entre el segundo semestre del año 2019 y el primer semestre del año 2019.

Se llevó a cabo la implementación de una estructura maestro-esclavo entre Python y Processing para la comunicación del control con la interfaz creada y desarrollada en una Raspberry Pi 3 B+, asímismo se presentan las pruebas lógicas de los sensores y encoders y mecánicas para la restauración mecánica como la ubicación en la posición cero y se desarrolló un sistema de control utilizando matrices cinemáticas dadas por Denavit-Hartenberg.

El control obtenido debe ser mejorado y se recomienda la implementación de otro controlador asímismo mejorar el sistema de potencia sustituyendo la fuente de poder por circuitos específicos para los sistemas implementados.

Palabras clave: Matrices cinemáticas, control, interfaz en python.

Abstract

This document aims to describe the technological update carried out at the Technological Institute of Costa Rica at the San Carlos regional headquarters to a SCORBOT ER-4U handling robot between the second semester of 2019 and the first semester of 2019.

The implementation of a master-slave structure between Python and Processing was carried out for the communication of the control with the interface created and developed in a Raspberry Pi 3 B+, as well as logical tests of the sensors and encoders and mechanics for the mechanical restoration as the location in the zero position and a control system was developed using kinematic matrices given by Denavit-Hartenberg.

The control obtained must be improved and the implementation of another controller is recommended as well as improving the power system by replacing the power source with specific circuits for the implemented systems.

Keywords: Kinematic matrices, control, interface in python.

Agradecimientos

Agradezco a mis padres y familia, que me han dado un apoyo incondicional.

Agradezco a todos mis amigos, quienes en el camino de este logro me han ayudado, me han dado palabras de aliento y fueron parte de mi lucha.

Agradezco a mis profesores, por darme la guía, probar mi conocimiento y enseñarme con tanta dedicación.

Agradezco a todos los que en el camino compartieron mi ilusión y me ayudaron a crecer personalmente.

Ruth Iveth Campos Artavia

Índice General

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

En este capítulo encontrará una descripción del problema existente, su contexto y la importancia del desarrollo de un sistema de control para el brazo robótico Scorbot ER-4U. Además incluye la descripción general de la solución desarrollada para lograr el control correspondiente y cumplir los objetivos.

1.1. Problema existente e importancia de su solución

El Instituto Tecnológico de Costa Rica (I.T.C.R) cuenta con un brazo robotico Scorbot ER-4U con 6 grados de libertad y una pinza controlada por un sistema neumático. Este robot manipulador tiene una amplia gama de aplicaciones en la industria como pintura, soldadura, ensamble, carga, descarga, entre otros y son los responsables de mejorar la calidad de vida de los humanos disminuyendo de manera paralela los costos de manufactura con un mayor grado de calidad, velocidad, capacidad, precisión asímismo es utilizado como herramienta para mejorar la educación de manera interactiva.

La problemática principal es la desactualización tecnológica del brazo robótico Scorbot ER-4U, ya que inicialmente contó con una unidad de control de la empresa Amatrol, la misma ya no se encuentra disponible en el equipo actual sino que posee un sistema de mando secuencial simple y por ello es necesario un control más robusto para lograr la programación de movimientos y secuen-

cias precisas implementando matrices cinemáticas. Adicionando a lo anterior, la institución no cuenta con fondos suficientes para restablecerlo por parte de la empresa Amatrol o adquirir otro con caracteristicas similares dado que es un equipo de alto valor monetario y su mantenimiento es poco asequible además de que ha sido descontinuado por parte de la empresa fabricante.

Dada la problemática actual, el proyecto pretende aprovechar los recursos e integración de dispositivos y procesamiento disponibles para mejorar el control existente y agregar funciones adicionales como control remoto del robot, restauración física del mismo y un entorno de programación que permita introducir comandos como el Melfa Basic 4 o similar para el control o posicionamiento del brazo en los puntos deseados.

La importancia de la solución radica en que al actualizar tecnológicamente y mejorar las características actuales del robot manipulador tanto su control y posicionamiento como su entorno de programación permitiría su utilización para actividades de investigación y docencia por parte de la institución.

1.2. Solución seleccionada

Dado que la resolución del problema planteado puede permitir la utilización del brazo robótico Scorbot ER-4U por parte de la institución, se acogieron los requerimientos establecidos por profesores interesados en utilizarlo para actividades formativas y futuras investigaciones. Por lo que en la tabla 1.1 se detallan los requerimientos formalizados por los funcionarios de la institución que se deben cumplir en la solución seleccionada a implementar.

Tabla 1.1: Requerimientos del sistema a implementar en la solución del proyecto.

ID	REQUERIMIENTO
1	La programación del sistema de control deberá ser implementado utilizando el lenguaje de programación Python o similar.
2	Se deberá utilizar como máximo una Raspberry o sistema similar para el control y procesamiento de datos
3	El sistema deberá tener una interfaz gráfica que permita el control tanto local como remoto.
4	El sistema no puede tener bugs, glitches o cualquier otra evidencia de un mal funcionamiento.
5	El sistema debe tener un mecanismo para recibir un archivo remoto y ejecutarlo correctamente.
6	El código fuente del sistema debe estar debidamente documentado, indicando funcionalidad, parámetros y restricciones para cada bloque de código
7	La interfaz debe iniciar un documento en blanco con comandos básicos de Melfa Basic 4 que sean seleccionado o introducidos por el usuario
8	El programa debe guardar los comandos introducidos en un documento de extensión ".txt" y debe ser capaz de revisar(compilar) para encontrar errores cuando es enviado remotamente y ejecutar en caso que no existan errores
9	Se deberá poder insertar puntos específicos para llegar a los mismos mediante interpolaciones no lineales, lineales, circulares calculadas con matrices cinemáticas
10	Se deberá utilizar una plataforma o programa existente que permita el envío de datos y actualización de los mismos para el intercambio de información entre el sistema del robot y el usuario remoto

Fuente: Elaboración propia

La solución implementada consta de 6 módulos, el primero es el módulo de la interfaz el cual le permite tener interacción con el usuario local, el segundo es el módulo de actuadores que está compuesto por los motores del robot, el tercero es el módulo de sensores, el cuál permite conocer la ubicación de cada motor mediante sensores y encoders, el cuarto módulo que lo conforma es el módulo de conexión a internet, este se encarga de proveer a los usuarios remotos el control efectivo del brazo robótico.

El quinto módulo es el control de potencia, este es el responsable de la alimentación eléctrica de cada circuito y finalmente el sexto módulo corresponde al módulo de procesamiento central, este se encarga de controlar la información del sistema para realizar tanto la habilitación de los motores y sensores, el procesamiento de la interfaz local y remota así como la conexión o desconexión de las fuentes de alimentación electrica y neumática. Se puede observar en la figura 1.1 un diagrama general de la solución planteada donde se muestran los distintos módulos.

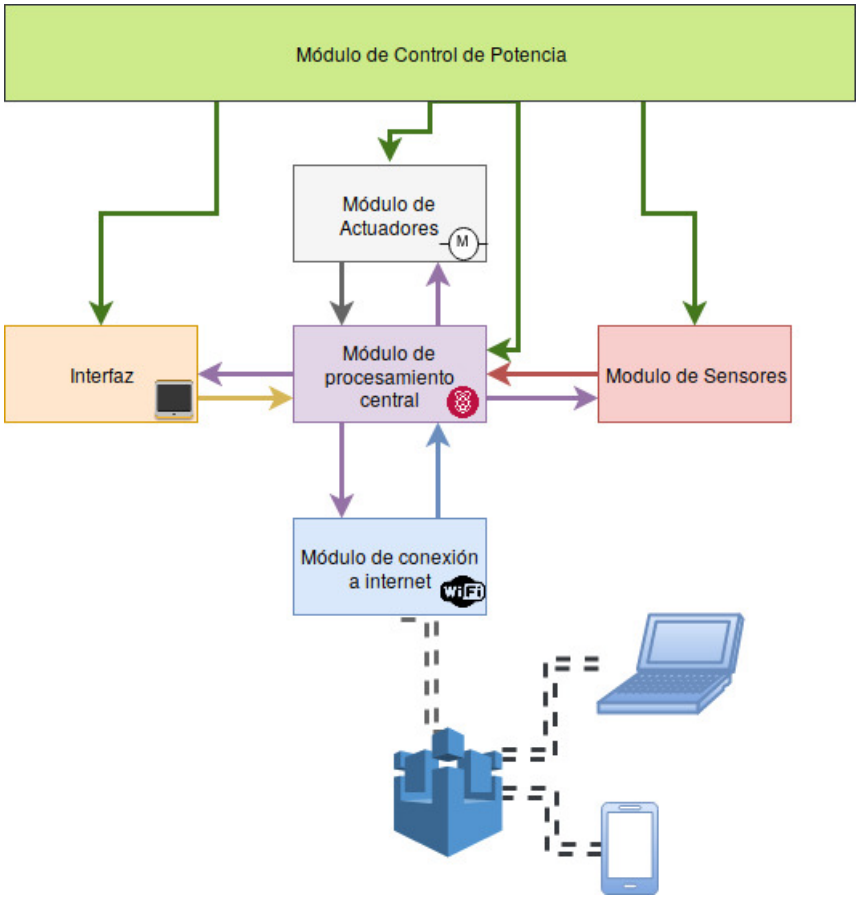


Figura 1.1: Diagrama de bloques general de la solución
Fuente: Elaboración propia

Capítulo 2

Meta y Objetivos

En este capítulo encontrará la meta y objetivo general que se persigue con el desarrollo y conclusión del proyecto. Además se indican los objetivos concretos para resolver el problema planteado y lograr una solución íntegra con las especificaciones.

2.1. Meta

Diseñar e implementar un sistema de control de posición y desplazamiento del Robot SCORBOT ER 4U ubicado en el Instituto Tecnológico de Costa Rica, mediante una conexión punto a punto a través de una interfaz local o remota, se logre programar, realizar desplazamientos y secuencias de movimientos de los grados de libertad.

2.2. Objetivo general

Desarrollar un sistema de control utilizando matrices cinemáticas para el brazo robótico SCORBOT ER 4U ubicado en el Laboratorio de Sistemas Digitales del ITCR-San Carlos.

2.3. Objetivos específicos

- Restablecer el sistema mecánico y de realimentacion de posición de manera que permita los desplazamientos precisos y medibles en cada uno de sus ejes.
- Desarrollar un programa de control con interfaz de usuario que permita la manipulación directa y la programación de secuencias a travez de comandos básicos basados en el lenguaje de programación Melfa Basic 4.
- Establecer un enlace punto a punto entre el brazo robótico ScorBot ER-4U y un usuario remoto para controlar el sistema de manera local.
- Redactar un documento de tipo paper para la publicación en una revista científica con formato IEEE Transactions.

Capítulo 3

Marco Teórico

En este capítulo encontrará una descripción del sistema a mejorar, algunos antecedentes bibliográficos de este tipo de sistemas, generalidades importantes para la resolución del problema asímismo principios físicos y/o electrónicos que están directamente relacionados con la solución del problema.

3.1. Descripción del sistema a mejorar

Ya que el proyecto será desarrollado alrededor de los robots industriales dado que se pretende dar solución a un problema en el brazo robótico Scorbot ER-4U se entenderá como robot industrial aquel manipulador multipropósitos reprogramable y controlado automáticamente en tres o más ejes [1]. Para proporcionar una solución certera, es necesario conocer los elementos básicos por los que se encuentra compuesto un robot industrial, entre ellos el sistema de accionamiento, sistema sensorial, sistema mecánico y sistema de control.[2]

Se entenderá como sistema mecánico al cuál está integrado por mecanismos que permitirán la correcta ejecución de tareas programadas, entre ellos actuadores, engranes, cadenas, poleas, ejes, tornillos, entre otros, los cuales son responsables de brindar transmisión al sistema, movimiento a los grados de libertad y control efectivo.[3]

La estructura mecánica del robot se encuentra formada por elementos o esla-

bones, los cuales están unidos entre ellos. Se referirán a articulaciones, aquellas que conectan una unión de entrada y una de salida y son capaces de permitir el movimiento relativo entre ellas. Las articulaciones pueden subdividirsen en lineales o rotatorias dependiendo del tipo de movimiento que ejecuten, dando una relación directa entre la unión de salida y la unión de entrada.[4] Se pueden visualizar los distintos tipos de articulaciones en la siguiente figura(ver la figura 3.1).

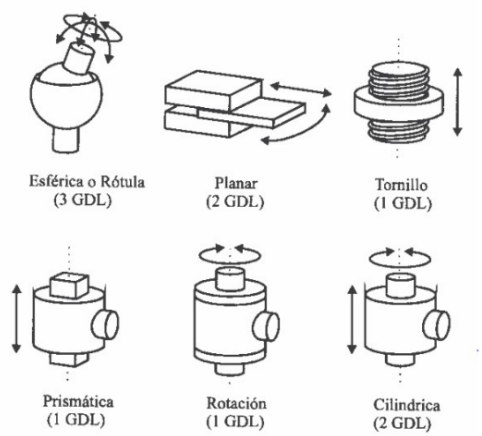


Figura 3.1: Tipos de articulaciones
Fuente: [2]

El sistema de accionamiento está integrado por actuadores y switches que permiten al robot ejecutar acciones según se requiera. Se refiere a actuadores aquellos que son capaces de generar el movimiento entre los elementos según la unidad de control y procesamiento de datos. Entre ellos se pueden subdividir en actuadores hidráulicos, neumáticos y eléctricos. Los dos últimos forman parte del sistema a actualizar por lo que es importante definir cada uno de ellos seguidamente: [5]

- Actuadores neumáticos: estos actuadores están basados en principios neumáticos para su funcionamiento. Se encargan de convertir la energía de aire comprimmido en movimiento mecánico.
- Actuadores eléctricos: su funcionamiento está basado en el uso de electricidad para la activación y movimiento, sin embargo tambien existen aquellos

que mediante una señal eléctrica dejan dar paso al aire comprimido o fluidos específicos.

Otro sistema importante en destacar es el sistema de transmisión,compuesto por elementos conocidos como transmisores encargados de transmitir el movimiento desde los actuadores hasta las articulaciones. Pueden convertir movimientos lineales en circulares y viceversa. Existen elementos reductores encargados de adaptar el par y velocidad de salida de los motores al movimiento de los elementos del robot.

El sistema sensorial al igual que en la estructura humana, le permite al robot conocer más de su entorno o tener referencia de ciertos eventos. Se denominan sensores aquellos instrumentos que permiten obtener una medición proporcional de la señal de las fuerzas o deformaciones basados en distintos fenómenos. Estos tienen una clasificacion muy diversa dependiendo de la aplicaciones de los mismos, por ejemplo los de contacto, ópticos, magnéticos, térmicos u otros. Seguidamente se detallan los sensores que son utilizados en el robot a actualizar y su función.

- Sensores de final de carrera: Estos son sensores de contacto, los cuales detectan el contacto entre el objeto y el sensor en cuestión.[6] Para el proyecto en cuestión, se utilizó un sensor o switch KW12-3 similar al que se encuentra en la figura 3.2, su principio de funcionamiento está basado en el de un switch, en el cual la salida correpoonde a la señal conectada al pin de normalmente abierto(NO), en caso contrario se obtiene en la salida la señal que posee el pin de normalmente cerrado(NC).

Figura 3.2: Representación del sensor de final de carrera colocado en la base deslizante del Scorbot ER-4U
Fuente: [7]

- Sensores de corte: Estos sensores cumplen la funcion de dar el posicionamiento inicial, en el proyecto fue utilizado el ITR8102(ver figura 3.4), el mismo cumple la función de un sensor óptico de barrera o herradura y es capaz de medir el desplazamiento y velocidad de cualquier motor cuando se encuentra en la posición deseada, se encuentra en el posicionamiento seteado como HOME, para ello se acopló a un circuito como se muestra en la siguiente figura(ver la figura 3.3). En este circuito se logra observar la conexión del ánodo a 3.3v, el cátodo conectado a tierra a través de una resistencia de 560 Ω, el colector a 3.3v y finalmente el emisor a tierra a través de una resistencia de 10k Ω, la salida se encuentra inmediatamente en la salida del emisor.

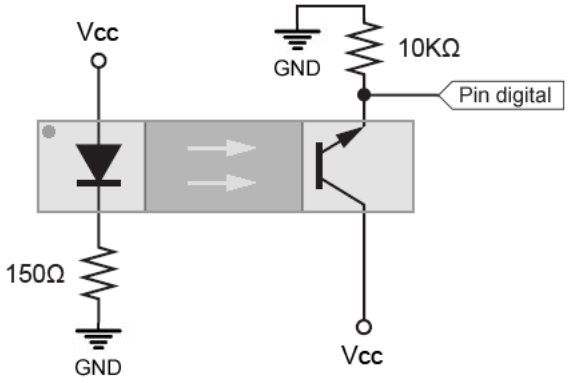


Figura 3.3: Circuito construido para cada uno de los sensores de posición del brazo robótico Scorbot ER-4U
Fuente: Elaboración propia

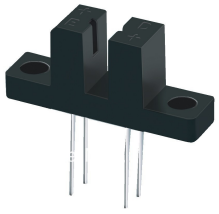


Figura 3.4: Sensor optoreflectivo ITR8102.
Fuente: [8]

- Encoder/Codificador: Un codificador es un dispositivo electromecánico que convierte el desplazamiento lineal o rotacional a señales digitales o de pul-

sos. Precisamente este es el codificador óptico más utilizado, implementado en la solución para control de robots (ver figura 3.5), consiste en un disco giratorio, una fuente de luz y un fotodetector (sensor de luz). El disco tiene patrones codificados y a medida que el disco gira proporcionalmente con el actuador, estos patrones interrumpen la luz emitida en el fotodetector, generando una señal de salida digital o pulsos dependiendo del posicionamiento en el disco.
No es necesario circuitos de pull-up o pull-down para el acople de las señales de salida debido a que la raspberry como sistema de control ya los tiene integrados en cada uno de sus pines de I/O disponibles.

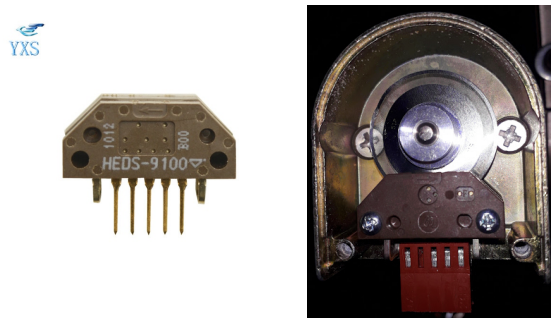


Figura 3.5: Representación del encoder utilizado para codificar los movimientos del robot Scorbot ER-4U.
Fuente: a)[9] b) Elaboración propia.

Adicionalmente es necesario que los robots cuenten con un sistema de control y manipulación del robot, con un controlador o el mismo sea diseñado partiendo del modelo dinámico del robot manipulador, este puede estar basado en conseguir que el robot o vehículo siga una referencia, se caracteriza porque las señales controladas son velocidades de rotación o direcciones de giro(control cinemático) o mediante aceleraciones de rotación y dirección tomando en cuenta sus dimensiones(control dinámico).[10]

El robot a actualizar corresponde a un Scorbot ER-4U, el cuál se caracteriza por ser un sistema robusto y versátil para educación y capacitación en el área de robótica industrial. El brazo robótico Scorbot ER-4U puede montarse en una base de mesa, pedestal o base deslizante lineal.[11] En el caso específico del proyecto, se tiene una base deslizante lineal que aumenta en un grado de libertad los movimientos del mismo.

De acuerdo con lo mencionado anteriormente, el brazo robotico con que cuenta el Instituto Tecnológico de Costa Rica tiene 6 grados de libertad con una pinza controlada por un sistema neumático y para el control y mejora del sistema se debe conocer las características del tipo de articulación y citar la forma en que serán definidas las articulaciones para dar solución al control. En la siguiente figura(ver figura 3.6) se puede observar los movimientos que es capaz de realizar el brazo robótico y según Kumar et all[12] se tiene un espacio como se muestra en la figura 3.7.

Figura 3.6: Movimientos del Scorbot para la definición del tipo de articulación.
Fuente: Elaboración propia

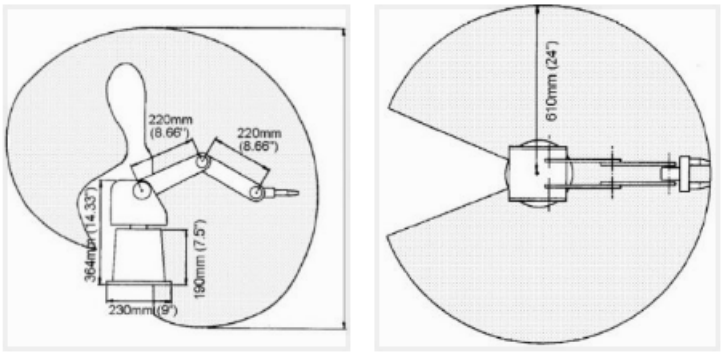


Figura 3.7: Espacio de trabajo del Scorbot ER-4U definido por la empresa fabricante.
Fuente:[12]

3.2. Antecedentes Bibliográficos

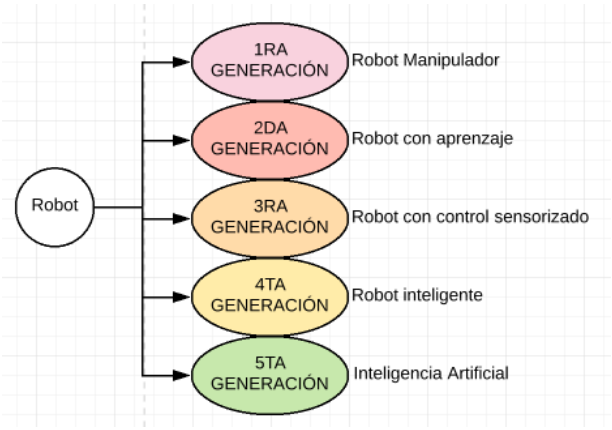
La última década se ha visto influenciada de manera importante de las aplicaciones industriales donde se automatizan los procesos mediante robots. La palabra robot fue acuñada del checo "robota" y se entenderá como robot a un manipulador programable con multiples funciones diseñado para mover materiales, partes, herramientas o dispositivos[1].

En el año 2010 Moslehpour, Odom, Barrett y Brown desarrollaron un sistema de control para el Scorbot ER III sustituyendo el control SCORBASE por un sistema de control mediante LabVIEW 8.0 para aumentar la eficiencia, estabilidad, simplicidad y utilizando el hardware inicial lo que permitió obtener un control completo de todas las variables y crear una interfaz amigable con el usuario. [13]

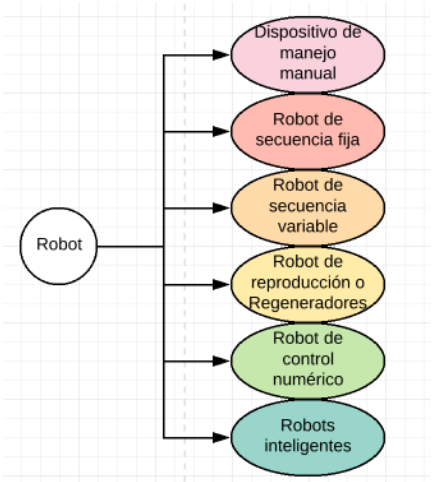
En el año 2011 se realizó un desarrollo de un sistema de control mediante Matlab a un Scorbot ER Vplus, este contempló la cinemática del robot, la cual se encontró a cargo de Verma y Deshpand, los mismos realizaron un modelado matemático del sistema según la convención establecida por Denavit-Harbenterg.[14]

Según Reyes existe una clasificación de robots de acuerdo con las áreas o funciones que cumplen, donde se resaltan los robots móviles(terrestres, marítimos y aéreos), humanoides(con un diseño complejo) y robots industriales(brazos robóticos y robots manipuladores).[1]

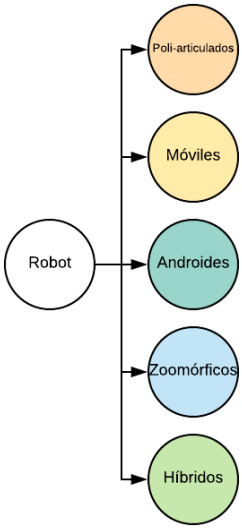
La clasificación y tipos de robots dependerá de la literatura donde se agrupan principalmente por generaciones, nivel de inteligencia (acuñada por la Asociación Japonesa de Robots(JARA)), la clasificación del Instituto de Robótica de América (RIA), nivel de control [15], nivel de lenguaje de programación, arquitectura, tipos de articulaciones, entre otros como se muestran citados en la figura 3.8.



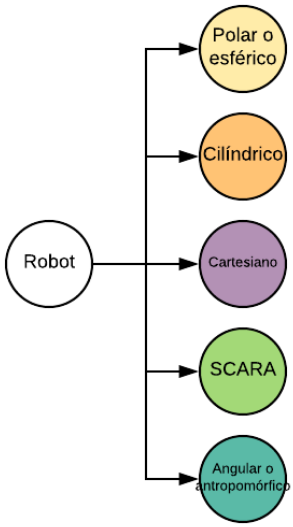
(a) Generaciones



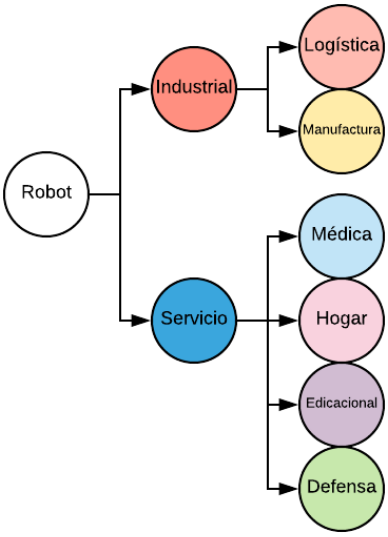
(b) Inteligencia



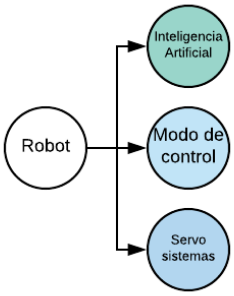
(c) Arquitectura



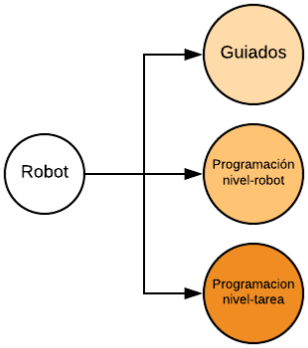
(d) Configuración



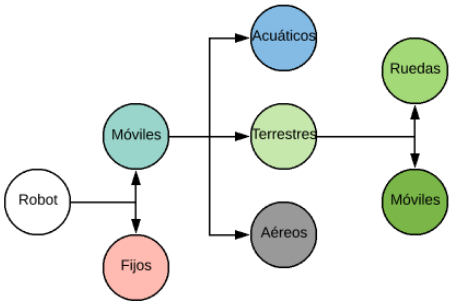
(e) Aplicación



(f) Control



(g) Lenguaje de programación



(h) Entorno

Figura 3.8: Clasificación de robots.
Fuente: Elaboración propia partir de [15]

3.3. Generalidades de robots

En esta sección se abordan conceptos importantes y generalidades de los robots que permiten una mejor especificación y selección del robot.

3.3.1. Especificaciones de robot

Inicialmente se tiene el concepto de *capacidad de carga*, que es la cantidad de masa que puede transportar el manipulador, en kilogramos, habitualmente los fabricantes proporcionan dicho dato. Los robots de cargas pequeñas oscilan alrededor de los 10kg a los 200kg, los de cargas medianas de 30kg a 800kg y los de cargas grandes o industriales de 150kg a los 1500kg de masa.[16]En este caso, el brazo robótico utilizado al tener según los fabricantes una capacidad de carga predeterminada es de 1 kg (2,2 lb) por defecto se clasificaría como un robot de carga pequeña, sin embargo puede ser capaz de mover 2,5 kg (5,5 lb) si se realiza una reducción de la velocidad de movimiento.

El término *movilidad* de un robot en robótica se refiere a la habilidad que tiene un robot móvil para desplazarse o moverse con libertad en su entorno. Esta se asocia de manera directa con la velocidad máxima a la que puede llegar cada motor y su potencial para frenado en el posicionamiento correcto[1]. La *velocidad máxima* es el valor máximo alcanzado normalmente con una carga de 0kg, se puede definir de manera independiente para cada articulación y depende del motor utilizado.[16]

Otra definición importante son los *grados de libertad* (GDL o DOF por sus siglas en inglés) de un robot, los cuales son obtenidos debido a la necesidad del posicionamiento de un objeto en el espacio. Usualmente para especificar el posicionamiento del objeto se utilizan tres coordenadas respecto a algún sistema de referencia (X , Y , Z) así mismo se utilizan tres ángulos para representar la orientación del sistema de referencia de los ejes respecto a otro sistema fijo, estos ángulos se conocen como ángulos de Euler(phi, theta, gamma), los cuales son representados en la figura 3.9, conocidos tambien como guiñada, cabeceo, alabeo (yaw, pitch, roll) [17]. Dado que el robot utilizado posee 5 ejes rotativos

y uno lineal de la base deslizante, por lo que con la pinza adjunta(6 GDL), se puede asegurar que el sistema es no redundante, porque el máximo de variables es de 6 y el número de grados de libertad es igual.[2]

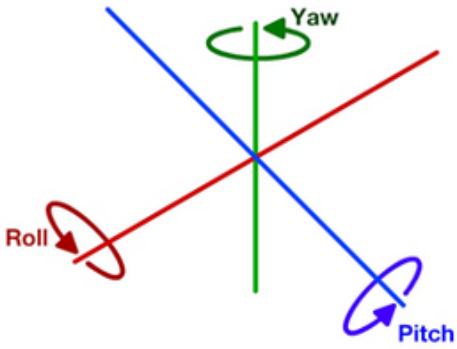


Figura 3.9: Representación de ángulos de Euler para un sistema tridimensional.
Fuente:
Fuente:[18]

A parte de lo anterior, se conoce como *espacio de trabajo(workspace) o zona de trabajo* de un robot manipulador al espacio o lugar donde el robot puede realizar todos sus posibles movimientos, definido como el grupo de puntos que pueden ser alcanzados por su efector final determinado por la geometría del robot, dimensiones y la naturaleza de sus articulaciones(lineales y rotacionales) donde los límites de giro de los motores y el desplazamiento restringen la zona de trabajo.

Existe una zona de trabajo primario, la cual está compuesta por todos los puntos que pueden ser accedidos desde distintas configuraciones de las articulaciones, asimismo una zona secundaria que se compone por los puntos que solamente pueden ser alcanzadas desde una orientación y comúnmente se encuentran lejanas a la base.

La forma, dimensiones y estructura del espacio de trabajo permite conocer el entorno donde trabajará el robot por lo que es importante caracterizar estos aspectos para la determinación del alcance del end-effector, asegurar la interacción correcta entre el robot y el entorno y ofrecer la posibilidad de optimizar las características del robot.

Algunos términos necesarios para la caraterización y resultados requeridos son:

Precisión(número de bits para representar una medición)[19].

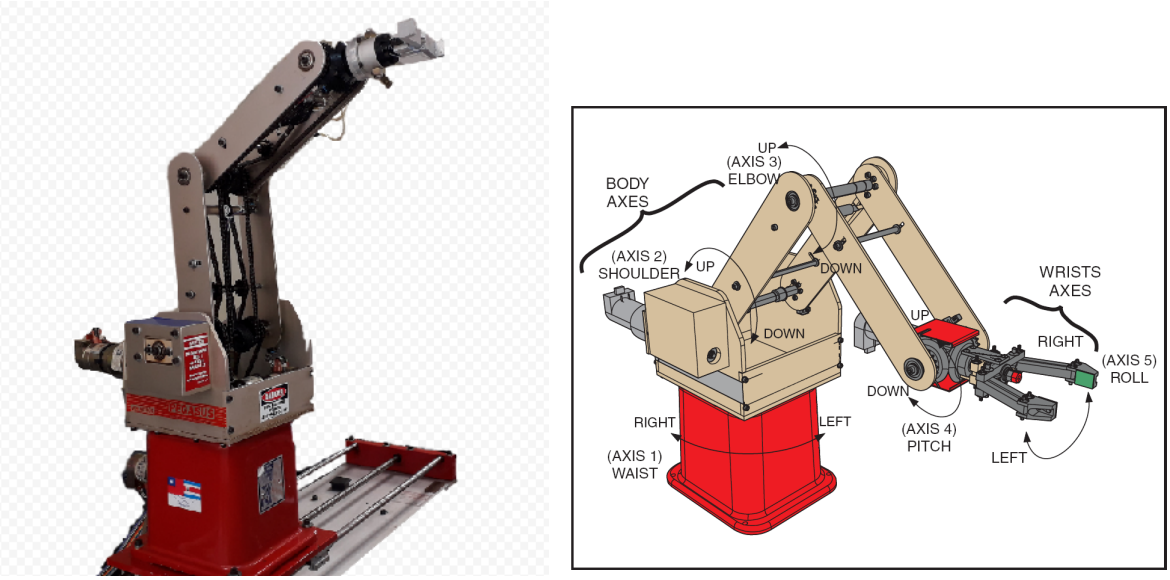
Resolución espacial(incremento más pequeño o mínima división que puede ser detectada o discernida en el espacio de trabajo entre dos puntos adyacentes), *inexactitudes mecánicas*(relacionadas con la calidad de los componentes que conforman las uniones y las articulaciones).[16] [19]

Exactitud(diferencia entre la medición realizada con respecto a la medición ideal o real y estará afectada por la masa de la carga dado que se genera resistencia mecánica y disminución de la velocidad nominal del motor).[16]

Repetitividad(capacidad de un robot en regresar al punto programado las veces que sean necesarias y es medido como la variación entre cada uno de los posicionamientos).[10]

3.3.2. Características del manipulador

El control del manipulador está basado en parámetros constantes de las dimensiones de cada uno de los elementos que constituyen el sistema mecánico, los cuales dependen de la geometría del manipulador como se muestran en la figura 3.10, en este caso los parámetros físicos geométricos del Scorbot ER 4U permiten la obtención de las ecuaciones del modelo mecánico del robot(ver Fig. 3.10(a) y 3.10(b) los parámetros del brazo robótico utilizado).



(a) Scorbot ER-4U del ITCR

(b) Movimientos del robot

Figura 3.10: Manipulador Scorbot ER-4U, se observan los distintos elementos que componen la estructura mecánica y las articulaciones que las unen.
Fuente: (a)Elaboración propia (b) [20]

Un robot con k-ésimos grados de libertad debe estar constituido por k-ésimos eslabones unidos por k-ésimas articulaciones, estos serán los parámetros geométricos. La longitud de las articulaciones son las distancias medidas entre sus ejes de rotación, estos se requieren para el estudio de las matrices cinemáticas.

Los motores utilizados en la solución corresponden al código GM9414J, tienen algunos parámetros establecidos por los fabricantes como un voltaje máximo de 30.3V, una intensidad de corriente máxima de 3A, Gear Ratios de 127.7:1, 65.5:1, 19.7:1, y 5.9:1(ver Tabla 3.1), acoplados a encoders de cuadratura HEDS-9100 Series I00, con un conteo de 512 ciclos por revolución(CPR)[21].

Tabla 3.1: Datos de caracterización de los motores utilizados.

Articulación	Tensión (V)	Corriente pico(A)	Corriente nominal (mA)	Velocidad (rpm)	Relación de transmisión
Base Deslizante	12	1.5	350	4810	5.9:1
Base	12	1.5	350	4810	19.7:1
Hombro	12	1.5	350	4810	19.7:1
Codo	12	1.5	350	4810	127.8:1
Pitch	12	1.5	350	4810	65.5:1
Roll	12	1.5	350	4810	65.5:1

Fuente: [22], [21]

La base deslizante del manipulador Scorbot ER-4U adiciona un grado de libertad lo que lo convierte en un robot de 6 GDL, por lo que permite tener un espacio de trabajo más extenso y aumenta las soluciones ante el acceso a determinados puntos. Esta base se comporta como una articulación lineal como se muestra en la tabla 3.2, la cual tiene una extensión de 925 mm esta posee acoplada una base de aluminio con 152,4 mm de ancho, 304,8 mm de largo y 76,2 mm de altura, esta base es el elemento que permite unir el robot manipulador Scorbot ER-4U a la base deslizante.[23]

Tabla 3.2: Tipo de movimiento e identificador según la articulación del brazo robótico Scorbot ER-4U.

Nº de Articulación	Articulación	Movimiento	Identificador
0	Base deslizante	Lineal	f
1	Base	Rotatorio	b
2	Hombro	Rotatorio	e
3	Codo	Rotatorio	a
4	Muñeca(pitch)	Rotatorio	d
5	Muñeca(roll)	Rotatorio	c

Fuente: Elaboración propia

El robot posee un elemento terminal, tambien conocido como pinza, End Effector o Gripper, este elemento se coloca en la muñeca del robot y es el encargado de interactuar con el medio. Sus características dependen de la naturaleza

del trabajo o tarea a realizar. En el caso específico de la solución, es utilizada para manipular elementos y trasladarlos a otros puntos dentro del espacio de trabajo como se muestra en la siguiente figura(abierta y cerrada respectivamente en la figura 3.11).

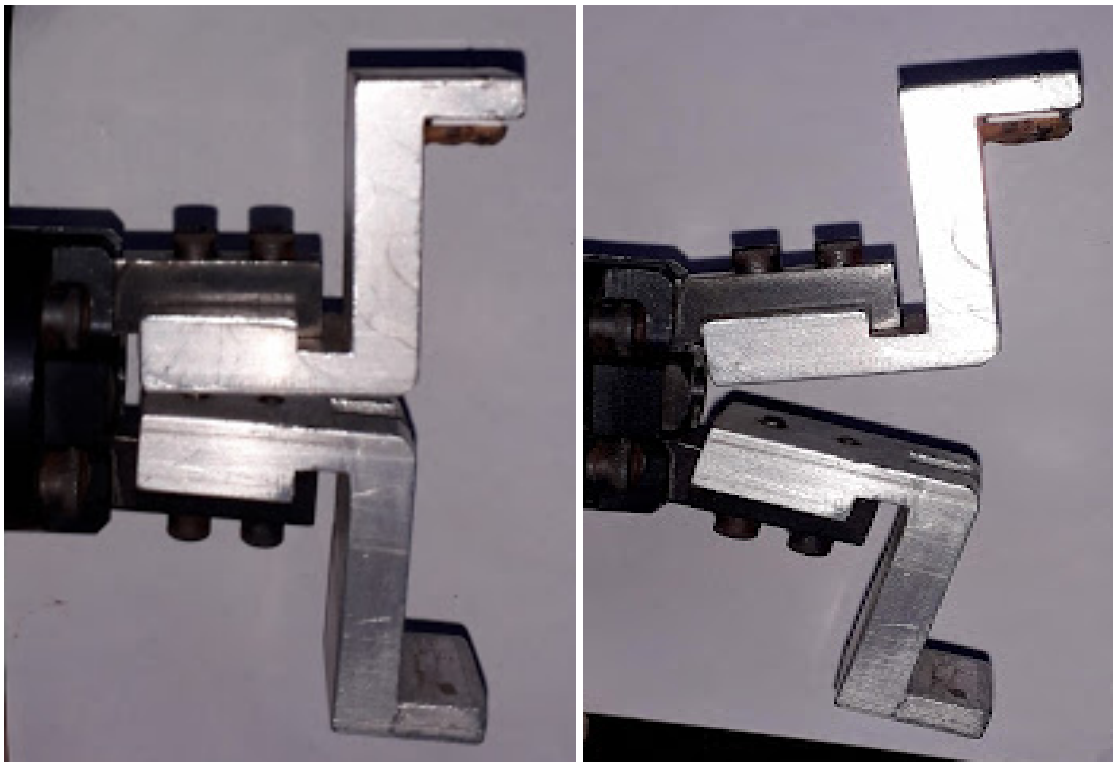


Figura 3.11: Sistema del elemento terminal o gripper
Fuente: Elaboración propia

3.4. Descripción de los principales principios físicos y/o electrónicos relacionados con la solución del problema

3.4.1. Análisis cinemático

En esta sección se expondrá el estudio del movimiento del robot Scorbot ER-4U respecto a un sistema de referencia de acuerdo con la configuración de los seis grados de libertad y la descripción del posicionamiento y orientación del elemento terminal con respecto a la base deslizante, para ello es necesario emplear la cinemática con el fin de realizar un estudio de la geometría del cuerpo, sin tener presente las fuerzas producidas por el movimiento, esta cinemática se subdivide en directa e inversa según el sistema de referencia.[24]

3.4.1.1. Cinemática directa

La función de la cinemática directa es determinar cuál es la posición y orientación del elemento terminal o pinza del manipulador respecto a un sistema de coordenadas seleccionado, partiendo de los valores de cada una de las articulaciones, además teniendo en cuenta los parámetros geométricos de los eslabones del robot mencionados anteriormente.[25]

Primeramente se necesita establecer la localización de cada uno de los eslabones de acuerdo con el sistema de referencia fijado y encontrar una matriz que permita relacionar la posición u orientación de cada eslabón (base, hombro,codo,pitch y roll respectivamente q1 q2 q3 q4 q5) o de la base deslizante(d1) con la orientación y posicionamiento de la pinza o end effector(x, y, z, alfa, beta, gamma) permitiendo conocer la ubicación del espacio tridimensional y con este asignar y determinar el posicionamiento de la pinza de una manera más efectiva.

3.4.1.2. Matrices de rotación

Las matrices de rotación permiten definir algebráicamente una rotación en un espacio 3D considerando un ángulo en el que está girando.

En la figura 3.12 se ejemplifica lo que hace una matriz de rotación, una vez proporcionado un ángulo hace que un sistema de referencia rote la cantidad de grados dado en una dirección convirtiendose en otro sistema distinto respecto al primero.

Figura 3.12: Rotación de un sistema de coordenadas un ángulo específico.
Fuente: Elaboración propia

La matriz de rotación para dos dimensiones se denota de la siguiente manera:

$$\begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \tag{3.1}$$

La matriz básica de rotación para un sistema espacial de tres dimensiones estará definido por el eje coincidente entre los dos sistemas OUVW y OXYZ, por ejemplo entre un sistema OUVW(Origen y ejes ortogonales UVW) y OXYZ(Origen y ejes ortogonales XYZ), si el eje coincidente es OU con OX, la matriz será como se muestra en la siguiente ecuación.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\alpha & -sin\alpha \\ 0 & sin\alpha & cos\alpha \end{bmatrix} \tag{3.2}$$

Si las rotaciones de los ángulos efectivamente son consecutivas, se puede representar la rotación global (alfa, phi, theta) así como se muestra en la figura 3.13.

$$\boldsymbol{T} = \boldsymbol{R}(x,\alpha)\boldsymbol{R}(y,\phi)\boldsymbol{R}(z,\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C\alpha & -S\alpha \\ 0 & S\alpha & C\alpha \end{bmatrix}\begin{bmatrix} C\phi & 0 & S\phi \\ 0 & 1 & 0 \\ -S\phi & 0 & C\phi \end{bmatrix}\begin{bmatrix} C\theta & -S\theta & 0 \\ S\theta & C\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{T} = \begin{bmatrix} C\phi C\theta & -C\phi S\theta & S\phi \\ S\alpha S\phi C\theta + C\alpha S\theta & -S\alpha S\phi S\theta + C\alpha C\theta & -S\alpha C\phi \\ -C\alpha S\phi C\theta + S\alpha S\theta & C\alpha S\phi S\theta + S\alpha C\theta & C\alpha C\phi \end{bmatrix}$$

Figura 3.13: Representación matricial de la rotación global en los tres ángulos de euler
Fuente: [26]

3.4.1.3. Matrices de transformación homogénea

La representación matricial permite considerar las transformaciones de un sistema de coordenadas a otro, donde se puede componer por matrices de distinto tamaño correspondientes a la rotación, traslación, perpectiva y escalado y la matriz resultante se conoce como matriz de transformación homogénea.

Esta matriz de transformación homogénea se utiliza para representar los eslabones, la posición y orientación de un sistema girado y trasladado con respecto a un sistema fijo de referencia, que es lo mismo que representar una rotación y traslación realizada sobre otro sistema de referencia.

3.4.1.4. Matrices de traslación

La matriz de traslación supone que el sistema únicamente cambia su punto de referencia tomando en cuenta los ángulos y parámetros del robot de un vector determinado con respecto al sistema inicial.

Para la obtención del modelo cinemático directo es necesario utilizar la perspectiva que ofrece la matriz de transformación homogénea, para ello existe un algoritmo comúnmente utilizado, llamado algoritmo de Denavit-Hartenberg que permite la resolución completa de la cinemática directa teniendo en cuenta las variables y parametros constantes.

3.4.1.5. Denavit-Hartenberg

De acuerdo con las reglas plantadas por Denavit-Hartenberg, se tiene que la matriz de transformacion final corresponderá a la aplicación de las distintas matrices de transformación como lo plantea González [16], mediante la aplicacion del algoritmo que se encuentra en la tabla 3.3.

Tabla 3.3: Tabla de las etapas que se deben seguir para obtener los parámetros según Denavit-Hartenberg

Etapa	Definición
D-H1	Numerar los eslabones comenzando con **1** (primer eslabón móvil de la cadena) y terminando con **n** (último eslabón móvil). La base fija del robot se numerará como eslabón **0**.
D-H2	Numerar cada articulación comenzando con **1** (la correspondiente al primer grado de libertad) y terminando con **n**.
D-H3	Localizar el eje de cada articulación. Si ésta es rotatoria, el eje será su propio eje de giro. Si es prismática, será el eje a lo largo del cual se produce el desplazamiento.
D-H4	Para **i** de **0** a $\mathbf{n-1}$ situar el eje $\mathbf{z_i}$ sobre el eje de la articulación $\mathbf{i+1}$.
D-H5	Situar el origen del sistema de la base $\{\mathbf{O_0}\}$ en cualquier punto del eje $\mathbf{z_i}$. Los ejes $\mathbf{x_0}$ e $\mathbf{y_0}$ se situarán de modo que formen un sistema dextrógiro con $\mathbf{z_0}$.
D-H6	Para **i** de **1** a $\mathbf{n-1}$, situar el sistema $\{\mathbf{O_i}\}$ (solidario al eslabón **i**) en la intersección del eje $\mathbf{z_i}$ con la línea normal común a $\mathbf{z_{i-1}}$ y $\mathbf{z_i}$. Si ambos ejes se cortasen, se situaría $\{\mathbf{O_i}\}$ en el punto de corte. Si fuesen paralelos, $\{\mathbf{O_i}\}$ se situaría en la articulación $\mathbf{i+1}$.
D-H7	Situar $\mathbf{x_i}$ en la línea normal común a $\mathbf{z_{i-1}}$ y $\mathbf{z_i}$.
D-H8	Situar $\mathbf{y_i}$ de modo que forme un sistema dextrógiro con $\mathbf{x_i}$ y $\mathbf{z_i}$.
D-H9	Situar el sistema $\{\mathbf{O_n}\}$ en el extremo del robot de modo que $\mathbf{z_n}$ coincida con la dirección de $\mathbf{z_{n-1}}$, y $\mathbf{x_n}$ sea normal a $\mathbf{z_{n-1}}$ y $\mathbf{z_n}$.
D-H10	Obtener θ_i como el ángulo que hay que girar en torno a $\mathbf{z_{i-1}}$ para que $\mathbf{x_{i-1}}$ y $\mathbf{x_i}$ queden paralelos.
D-H11	Obtener $\mathbf{d_i}$ como la distancia, medida a lo largo de $\mathbf{z_{i-1}}$, que habría que desplazar $\{\mathbf{O_{i-1}}\}$ para que $\mathbf{x_i}$ y $\mathbf{x_{i-1}}$ quedasen alineados.
D-H12	Obtener $\mathbf{a_i}$ como la distancia medida a lo largo de $\mathbf{x_i}$ (que ahora coincidiría con $\mathbf{x_{i-1}}$) que habría que desplazar el nuevo $\{\mathbf{O_{i-1}}\}$ para que su origen coincidiese con $\{\mathbf{O_i}\}$.
D-H13	Obtener α_i como el ángulo que habría que girar en torno a $\mathbf{x_i}$ (que ahora coincidiría con $\mathbf{x_{i-1}}$), para que el nuevo $\{\mathbf{O_{i-1}}\}$ coincidiese totalmente con $\{\mathbf{O_i}\}$.
D-H14	Obtener las matrices de transformación ${}^{\mathbf{i-1}}\mathbf{A_i}$.
D-H15	Obtener la matriz de transformación entre la base y el extremo del robot $\mathbf{T} = {}^{\mathbf{0}}\mathbf{A_1} \cdot {}^{\mathbf{1}}\mathbf{A_2} \cdot {}^{\mathbf{2}}\mathbf{A_3} \ldots {}^{\mathbf{n-1}}\mathbf{A_n}$.
D-H16	La matriz **T** define la orientación (submatriz de rotación) y posición (submatriz de traslación) del extremo referido a la base en función de las **n** coordenadas articulares.

Fuente: [27]

3.4.2. Control

En esta sección se citan los distintos controladores disponibles para dar solución al control cinemático del brazo.

3.4.2.1. Controlador

Este es un dispositivo que permite la lectura de todas las señales a controlar y da respuestas empleando un determinado programa o algoritmo

- Controlador de Lógica Programable(PLC): Este controlador posee la cualidad de ser eficiente, tiene terminales de salida y de entrada, las de salida son aquellas que permiten controlar el sistema de manera secuencial mientras que las terminales de entrada serán las que provean datos de realimentación para lograr un control completo y eficiente. [28]
- Raspberry Pi 3: Es un modelo de tercera generación de la marca Raspberry pi, cuenta con características como 1GB RAM,Procesador Quad Core 1.2GHz Broadcom BCM2837 y GPIO extendido de 40 pines que permite la medición y control de variables además de conexión remota al mismo mediante programación en alto nivel y/o programas [29]
- Arduino: Este controlador posee un microcontrolador AVR ATmega328 de paquete doble en línea (DIP) extraíble. Tiene pines de entrada / salida digital (de los cuales pueden usarse como salidas PWM y otros como entradas analógicas). Los programas se pueden cargar desde el programa informático Arduino, es un dispositivo de adquisición de datos y logica programable en pseudo c, lo que permite obtener datos en tiempo real y dar respuestas remotas mediante el uso modular.

3.4.2.2. Lenguajes de programación

En esta sección se mencionan los distintos lenguajes de programación y librerías utilizadas para el control del brazo robótico.

- Python: es un lenguaje de programación interpretado(interpreta el código) y multiplataforma (funciona en muchos sistemas operativos). Este permite la programación en texto plano(con un editor de texto) para ser utilizado tanto en windows como linux.[30]

- Laboratorio de matrices (Matlab acrónimo en inglés): Es un software desarrollado por MathWorks que permite la programación y manipulación de matrices, trazado de funciones y datos, implementación de algoritmos, creación de interfaces de usuario e interconexión con programas escritos en otros idiomas, incluidos C, C ++, C #, Java, Fortran y Python.[26]

- Processing, mode Java: Processing es un software flexible, posee un lenguaje para aprendizaje basado en java y permite la producción de proyectos audiovisulaes. En el 2001, Processing ha incluido tecnología visual con el fin de mostrar gráficos instantáneos y visual de la información.

Librerías disponibles en python

En esta sección se retoman las librerías disponibles y necesarias para dar solución al problema de desactualización del brazo robótico.

- Librería numpy: Es un paquete básico para la resolución de problemas científicos con Python. Contiene una matriz N-dimensional, funciones sofisticadas (difusión), herramientas para integrar el código C / C ++ y Fortran, álgebra lineal útil, transformada de Fourier y capacidades de números aleatorios. NumPy también se puede usar como un contenedor multidimensional eficiente de datos genéricos. Se pueden definir tipos de datos arbitrarios. Esto permite a NumPy integrarse de manera rápida y sin problemas con una amplia variedad de bases de datos. NumPy está licenciado bajo la licencia BSD, lo que permite su reutilización con pocas restricciones.[31]

- Librería Sympy: es una biblioteca Python para matemática simbólica. Su propósito es convertirse en un completo sistema de álgebra computacional que pueda competir directamente con alternativas comerciales (Mathematica, Maple) manteniendo, a la vez, el código tan simple como sea posible

para hacerlo extensible de manera fácil e integral. SymPy está escrito completamente en Python y no necesita usar otras bibliotecas. [32]

- Librería Math: Este módulo está siempre disponible para Python. Proporciona acceso a las funciones matemáticas definidas por el estándar C. Estas funciones no pueden usarse con números complejos. Excepto cuando se indique explícitamente lo contrario, todos los valores de retorno son flotantes. [33]

- Librería Threading: En programación el término "multithreading"se refiere a la técnica que permite que una aplicación ejecute simultáneamente varias operaciones en el mismo espacio de proceso. A cada flujo de ejecución que se origina durante el procesamiento se le denomina hilo(thread en inglés) o subproceso, pudiendo realizar o no una misma tarea. En Python, el módulo threading hace posible la programación con hilos. [34]

- Librería GPIO:Este paquete proporciona una clase para controlar el GPIO en una Raspberry Pi. Se debe tomar en cuenta que este módulo no es adecuado para aplicaciones de tiempo real o críticas ya que no se puede predecir cuándo estará ocupado Python recolectando basura. También se ejecuta bajo el kernel de Linux, que no es adecuado para aplicaciones en tiempo real: es un sistema operativo multitarea y otro proceso puede tener prioridad sobre la CPU, lo que causa fluctuaciones en su programa.[35]

- Librería MCP: es una pseudo librería que permite la habilitación de hasta 128 pines del MCP23017 mediante el uso de los dos pines I2C que posee la raspberry pi 3 model B+, es utilizado para conectar todo tipo de dispositivos adicionales (leds, botones...) y el control de señales.[36],[37]

3.4.2.3. Interfaz de conexion remota

En esta sección se expondrán las distintas interfaces y protocolos que se utilizarán para la resolución del problema y que harán posible la conexión remota entre dispositivos con el sistema de control del brazo robótico Scorbot ER-4U.

- Computación de red virtual (Virtual Network Computing-VNC): Cuando no es conveniente trabajar directamente en la Raspberry Pi es posible trabajar desde otro dispositivo controlando remotamente las acciones y el sistema Raspbian incluye el paquete de control VNC. VNC es un sistema de escritorios gráficos compartidos y posee una interfaz que permite el control remoto desde otra computadora o movil(VNC Server), transmitiendo las acciones del teclado, mouse o los eventos táctiles al servidor VNC, y recibe actualizaciones en la pantalla a cambio(VNC Viewer).[38]
- SSH (Secure Shell): Es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente-servidor y permite a los usuarios conectarse a un host remotamente. SSH cifra toda la sesión de conexión proporcionando alta seguridad.[39]
- AWS IoT Greengrass: Es un servicio ofrecido por Amazon Services que se utiliza en los dispositivos para que puedan actuar localmente en los datos que generan, mientras siguen usando la nube para administración, análisis y almacenamiento. Además los dispositivos conectados pueden ejecutar predicciones basadas en modelos de aprendizaje automático, mantener sincronizados los datos del dispositivo y comunicarse con otros dispositivos de forma segura, incluso cuando no están conectados a Internet.[40]

Capítulo 4

Procedimiento metodológico

En este capítulo encontrará una descripción detallada de las etapas del método de diseño de ingeniería para obtener la solución del problema que fue más acorde con las limitaciones y permitiera cumplir los requerimientos establecidos.

4.1. Reconocimiento y definición del problema

Primeramente se debió reconocer el problema general de la desactualización del brazo robótico, luego se procedió a seccionar el mismo en subproblemas y para definirlos se realizó un chequeo general del sistema, con ello se encontraron varias deficiencias como se muestra a continuación:

- Interfaz gráfica robusta inexistente por lo no posible el control directo del brazo robótico.
- Mando a distancia del sistema faltante, lo que no permite la programación remota ni visualización del mismo.
- No contó con interfaz para la programación en lenguaje de manipuladores.
- Movimiento de transmisión de los motores ineficiente por control inexistente y desajuste mecánico.

- No existía una adquisición ni procesamiento de los datos, ni la lectura de las señales de sensores.
- No existe un cableado eléctrico ni cálculo de potencia acorde con la solución y control del brazo.

Por lo anterior, se planteó resolver el problema desde cinco aristas distintas: interfaz, control, eléctrico, mecánico y comunicación. Después de realizar la identificación de subproblemas, se realizó un levantamiento de metas y restricciones de la solución que se pretendían cumplir mediante el desarrollo del proyecto, entre ellas se planteó restablecer el sistema eléctrico y mecánico del modelo inicial para lograr una lectura adecuada y correcta de los datos proporcionados por el sistema sensorial y disminuir la variación mecánica del brazo robótico. Otra de las metas planteadas fue lograr la conexión de un usuario remoto al sistema de control mediante una plataforma o visor de escritorios de tal manera que se pudiera programar remotamente los comandos de movimientos, ejecutar un control directo sobre el robot y enviar, verificar y ejecutar un documento con comandos elegidos del lenguaje compilador elaborado desde otro dispositivo.

Adicionalmente se buscó desarrollar una interfaz de programación intuitiva y amigable con el usuario, que permitiese tanto control directo como programado de los comandos básicos que fueron seleccionados de Melfa Basic 4, este último fue elegido dado que es un lenguaje actualizado básico para la programación de manipuladores. De manera análoga se buscó producir de manera programada un sistema de control basado en matrices cinemáticas del robot para el posicionamiento de la pinza en el punto respectivo con el fin de utilizar los parámetros del robot y las restricciones mecánicas del mismo.

Después de plantear las metas se realizó un análisis para encontrar las restricciones de la solución y entre ellas se obtuvo que el proyecto debía ser elaborado con un presupuesto limitado de tal manera que se encontrara una solución de bajo presupuesto. Otra restricción planteada fue utilizar las herramientas, dispositivos y materiales actuales que posee el Instituto Tecnológico de Costa Rica para dar una solución de un sistema rígido y eficaz ya que otra limitante es el tiempo para obtener componentes y dispositivos para el control actualizado.

4.2. Obtención y análisis de información

Inicialmente se realizó una investigación preliminar, la misma constó en obtener información del estado actual del brazo robótico con el profesor coordinador quién dispuso información de un manual del control actual del mismo elaborado por estudiantes de la escuela de Ingeniería Electrónica. El profesor dió algunas pautas a seguir para el control del mismo de tal manera que se realizara la categorización y elección de la información a utilizar para el diseño del control de la mejor manera mediante libros pertinentes y papers.

Además de eso se realizó un seguimiento al manual elaborado por los estudiantes que estuvieron a cargo del primer control del robot donde ellos señalaron varias limitaciones que presentaba el control, con esto finalizó la etapa preliminar. Continuamente se realizó un análisis del control actual para comprender, dada la información obtenida, las problemáticas que este poseía y plantear posibles soluciones. En conjunto con la investigación preliminar o método cualitativo se llevó a cabo una verificación cuantitativa donde se realizaron mediciones y aplicando pruebas lógicas para obtener información a partir del robot y su funcionamiento particular permitieron la comprensión del problema.

A posteriori se realizó una búsqueda de los documentos y materiales técnicos y bibliografía pertinente para obtener información sobre los dispositivos disponibles, número de ellos, localización, facilidad de compra, envío e implementación en la solución, las características necesarias para la implementación de distintas soluciones de interfaz o de lectura de datos y aumento de pines, capacidad actual y capacidad máxima de almacenamiento y procesamiento, paralelamente se realizó una búsqueda bibliográfica y selección de la misma mediante una categorización donde se buscó y su confiabilidad primero se buscó información sobre el robot realizada por la empresa fabricante después se leyeron aquellos papers o documentos que poseían información sobre el sistema del robot ya que la empresa fabricante no proveía datos necesarios para la comprensión total del sistema, de la misma forma se buscó una manera de verificación de los dispositivos eléctricos o la mecánica que componen al robot.

La información que se encontró en el libro de Reyes permitió ampliar la

visión sobre el control y cómo debía efectuarse, se analizaron problemas mecánicos seguidamente se buscó en la literatura la manera en la que se debía solucionar los problemas mecánicos enfrentados para mejorar la precisión de un brazo robótico. En ella se daban algunas recomendaciones por ejemplo desensamblar las cadenas y sistemas mecánicos para encontrar algún desperfecto del mismo al realizar esto no se encontró por lo que se obtuvo información de documentos bibliográficos con el fin de solidificar el conocimiento mecánico de robots respecto a las cadenas, engranes, tensores y tipos de configuraciones.

Adicionalmente se realizó investigación acerca de las distintas soluciones de control a través de matrices cinemáticas existentes. Tambien se investigó acerca del análisis necesario para la aplicacion de la cinemática así como el control de velocidades y los diferentes indices de desempeño para la medición de la repetitividad de los movimientos del robot, entre esto se encontró que las matrices cinemáticas es una de las soluciones convencionales para robots que poseen estructuras de uno hasta 5 grados de libertad por lo que es pertinente para el control de robots.

Para analizar la información obtenida que permitiría la implementación en la solución final se elaboraron varias tablas de comparación que permitieron obtener mejor visualización de los datos obtenidos y según las mismas, una solución que presentara mejor desempeño en términos de procesamiento, interfaz, conexión, mecánico, de potencia y bajo costo de la solución. Algunos objetivos propuestos para disminuir el tiempo de implementación fueron hacer uso de los 6 motores del robot y los drivers L298N, además los encoders integrados a cada motor, una pantalla tactil 7” y los sensores ópticos de cada uno de los grados de libertad por lo que fue limitado por los dispositivos que se pretendieron utilizar.

4.3. Evaluación de las alternativas y síntesis de una solución

La solución se dividió en cada uno de los módulos propuestos para lograr cada uno de los objetivos y meta definidas inicialmente en cada una de las partes que integran la solución, se revisó en términos de principios de funcionamiento y lógica y se realizó una retroalimentación después de la aplicación de distintas

pruebas para establecer si el cumplimiento de cada requisito del módulo y al probar cada etapa, se encontró la solución más acertada que permitiera desarrollar la solución de la manera más factible y económica.

Dado que el problema fue abordado desde 5 aristas distintas se tomó en cuenta las alternativas en cada una de estas. Primeramente respecto a la creación del sistema de control se tomaron en cuenta tres propuestas para los controladores que se detallan en la tabla 4.1 y se muestra una comparación de las características de los distintos controles posibles para que la selección del más apto para la solución se realizara de acuerdo con las características más importantes y limitaciones para la elección del controlador,las propuestas son la implementación de una Raspberry Pi,un sistema PLC-Computador y un sistema con Arduino. Se escogió como alternativa de solución el uso de la Raspberry Pi, por varios motivos, entre ellos, la flexibilidad a cambios, la conexión remota y la posibilidad de interfaz además de ello que se presupuestó el tiempo para aprender sobre el control mediante este tipo de controlador. De acuerdo con esta solución entonces se discriminaron la cantidad de tipos de comunicación que se podrían implementar dado que la Raspberry Pi tiene un sistema operativo interno y sería utilizado para el control dado que la interfaz también permitiría la conexión del usuario remoto con la raspberry pi y el sistema de visor de escritorio.

Tabla 4.1: Tabla comparativa de las características de tres distintas propuestas para el controlador de la solución.

Características del controlador	Propuesta 1:RASP	Propuesta 2: PC-PLC	Propuesta 3: Arduino
Conexión Remota(SI)	SI	NO	SI
Flexibilidad a cambios (ALTA)	ALTA	BAJA	MEDIA
Programación (FÁCIL)	DIFICIL	FÁCIL	FÁCIL
Navegación (FÁCIL)	FÁCIL	MEDIA	DIFICIL
Velocidad de procesamiento de datos (ALTA)	ALTA	BAJA	BAJA
Costo (BAJO)	BAJO	ALTO	ALTO
Potencia(MEDIA)	MEDIA	BAJA	MEDIA
Sistema Robusto(MEDIA)	MEDIA	ALTA	BAJA
Interfaz (SI)	SI	SI	SI

Fuente: Elaboración propia

Nota:El parámetro entre paréntesis denota la característica deseable

Se presenta con anterioridad la tabla 4.1, se agrega un análisis adicional para la selección de las propuestas de solución al sistema general donde la propuesta 1 es la seleccionada ya que los materiales se encuentran disponibles en la insti-

tución. La propuesta dos se descarta ya que no se tendría una conexión remota para su control además su valor monetario excede el presupuesto actual y la propuesta 3 se descarta dado que se necesitaba comprar los componentes para la interfaz, interconexión y brindaría un sistema simple y que no permitiría la implementación de matrices cinemáticas y el procesamiento correcto de los datos para la retroalimentación.

En la tabla 4.2 se muestran las propuestas para los sistemas de comunicación entre ellos se tenía la primera propuesta que era una comunicación por ssh la cual se vio discriminada dado que no permitía una interfaz hacia el usuario remoto por lo que no se cumplía con una comunicación efectiva visual, seguidamente se tenía la propuesta de utilizar el sistema de Amazon para utilizar los servidores pero la implementación incluía un valor monetario que no se contaba, ya que una de las limitantes que se definió en el proyecto fue encontrar una resolución del proyecto con bajo costo y bajo presupuesto.

Tabla 4.2: Tabla comparativa de las características de tres distintas propuestas para la conexión del usuario remoto.

Aspectos	Propuesta 1: ssh	Propuesta 2: Amazon Services	Propuesta 3: VNC
Necesita internet(NO)	NO	SI	SI
Comparte Imagen (SI)	NO	NO	SI
Compatibilidad Raspbian (SI)	NO	SI	SI
Gratuito (SI)	SI	NO	SI
Control Remoto del Escritorio (SI)	NO	NO	SI
Seguridad(ALTA)	BAJA	ALTA	MEDIA
Trasmisión de archivos(SI)	SI	SI	SI

Fuente: Elaboración propia

Nota:El parámetro entre paréntesis denota la característica deseable

En la siguiente tabla se observa la comparación entre tres propuestas para el sistema de potencia (ver tabla 4.3), entre ellas una batería, una fuente de poder o un panel solar con una batería, la fuente de poder fue la solución seleccionada ya que presentaba un suministro de varias tensiones además de que se encontraba disponible para su implementación pero la configuración inicial debía ser modificada.

Tabla 4.3: Tabla comparativa de propuestas para el sistema de potencia de la solución. Elaboración propia

Sistema Potencia	Propuesta 1 Batería	Propuesta 2 Fuente de Poder AT	Propuesta 3 Panel solar con batería
Ahorro (SI)	SI	NO	SI
Suministro de distintas tensiones(SI)	NO	SI	NO
Disponibilidad(SI)	NO	SI	NO

Fuente: Elaboración propia

Nota:El parámetro entre paréntesis denota la característica deseable

El tercer sistema implementado fue el eléctrico en la que se planteó utilizar el cableado actual de manera arbitraria, este se escogió dado que ya se encontraba funcional a pesar de que visualmente no era estético, se eligió seguir haciendo uso de la fuente de poder dado que las tensiones suministradas eran amplias para la implementación de los sensores, los encoders y motores, no se eligió la batería externa o el panel solar dado que los mismos primero no se encontraban disponibles y segundo no hay una variedad de voltajes que si se tienen con la fuente poder que se está utilizando actualmente. Se descartó una cuarta propuesta que era la implementación de un circuito a un voltaje de 12 voltios y que a partir de este circuito se obtuvieran los voltajes menores pero limitaba la cantidad de corriente suministradda, el cuál no se desarrolló ni se propuso por la falta de componentes y tiempo.

Respecto al sistema sensorial se eligió utilizar los sensores, motores y codificadores actuales dado que los mismos podrían ser funcionales para dar la solución adecuada ya que son los mismos fueron utilizados en el control de la empresa fabricante y en cuestion de la adquisición de datos e implementación de un sistema de sensado independiente del control en la cual era posible la evaluación de algunas propuestas mediante el uso de una arquitectura maestro-esclavo para el analisis de datos y control del robot, de estas se propone un sistema remoto o local con raspberry pi, arduino, el primero se basaba en la conexión de una raspberry pi para adquirir los datos y enviarlos a una compuradora para el control y análsis de datos, la segunda propuesta se basaba en utilizar un sistema local en la raspberry donde se hace tanto la adquisición como el uso de los datos y el controlador, la tercera propuesta era utilizar un arduino para la adquisión de datos y la conexión de una computadora personal para el analisis de datos como

se muestran en la tabla 4.4. De la tabla anterior se decidió utilizar la raspberry pi local para no restringir el uso de cualquier usuario que no contase con un computador extra o capacidades tecnológicas que con el tiempo podrían variar o no estar disponibles.

Tabla 4.4: Tabla comparativa de propuestas de los distintos sistemas de sensado a implementar en la solución.

Características del sistema de sensado	Propuesta 1 Remoto Raspberry	Propuesta 2 Local Raspberry	Propuesta 3 Remoto Arduino
Tiempo de procesamiento(BAJO)	ALTO	BAJO	MEDIO
Complejidad (MEDIA)	ALTA	MEDIA	FÁCIL
Programación(PYTHON)	C	PYTHON	C++
Velocidad de comunicación(RÁPIDA)	MEDIA	RÁPIDA	LENTA
Dependencia de red (NO)	SI	NO	SI

Fuente: Elaboración propia

Nota:El parámetro entre paréntesis denota la característica deseable

En el siguiente cuadro comparativo(Tabla 4.5) se puede observar las distintas propuestas para la solución de la interfaz, la primera de ellas Tkinter, la cuál posee una gran desventaja debido a la velocidad de interfaz y la no posibilidad de simulación, la propuesta tres, consta de utilizar Matlab, la cual posee posibilidad de simulación y una alta velocidad y finalmente la plataforma de Processing como la propuesta menos viable y aunque su complejidad no es alta, posee una mejor simulación de gráficos y velocidad en el sistema de control elegido por lo que al comprometer la velocidad del procesador se eligió la primera de ellas.

Tabla 4.5: Tabla comparativa de propuestas para el sistema de interfaz de la solución.

Interfaz	Propuesta 1 Python Tkinter	Propuesta 2 Processing	Propuesta 3 Matlab
Complejidad(BAJA)	BAJA	MEDIA	ALTA
Velocidad (MEDIA)	LENTO	MEDIA	RÁPIDO
Simulación y Gráficos(SI)	NO	SI	SI
Compatibilidad Raspbian(SI)	SI	SI	NO

Fuente: Elaboración propia

Nota:El parámetro entre paréntesis denota la característica deseable

La solución sintetizada contará con las siguientes secciones, las cuales abordan las distintas etapas en que será subdividida la solución:

- Módulo de Interfaz: El sistema tiene la capacidad de servir como interfaz local mediante una aplicación diseñada para el uso de la pantalla de 7"para el control del brazo robótico. Debe permitir la capacidad de programar al usuario local o remoto como desplegar la información importante paa el mismo, transmitir, verificar, crear y editar documentos con comandos para la programación y control del robot.

- Módulo de actuadores: La solución implementa motores DC acoplados a drivers de motores como actuadores que son los dispuestos por la empresa fabricante en el robot, los cuales requieren una alimentación en corriente directa para el funcionamiento y movimiento en cada uno de los ejes principales así como el desplazamiento en el eje de la base.

- Módulo de sensores:

  - Codificadores ópticos: Se implementan codificadores ópticos incrementales de dos canales en la solución para traducir la posición rotatoria de cada uno de los motores DC.
  - Sensores de punto cero: Se emplean sensores o interruptores de final de carrera de tipo optoreflectivo o de contacto para llevar a la posición inicial o cero cada uno de los motores que componen las articulaciones del brazo robótico con el fin de conocer si ya se encuentran en la posición establecida como cero o home.

- Módulo de procesamiento central: Este módulo es capaz de proporcionar control general del sistema así como la conexión de los distintos módulos con internet. Estará compuesto por:

  - Microcontrolador: El microcontrolador empleado proporciona una conexión a internet, la conexión a una pantalla para despliegue de datos del módulo de interfaz así como lectura de las instrucciones de un documento de texto para su ejecución.
  - Mandos: Los mandos implementados son compatibles con las tareas a realizar, ya sea mediante switches para el encendido y apagado del robot y touchpad para la ejecución de las instrucciones y desplazamientos a las posiciones definidas, estos mandos están dispuestas en

un panel de notificación que permite conocer el estado de cada uno de los motores y estado general del robot.

- Módulo de control de potencia: Se hizo uso de una batería para el control de potencia para dotar de funcionamiento a los motores al ejecutar las instrucciones así como alimentación a los distintos sensores del sistema robótico.
- Módulo de comunicación: Se implementó una conexión inalámbrica para la conexión del usuario remoto y que el mismo realice peticiones sobre el control creado.

El sistema en conjunto proporcionaría un control efectivo del brazo robótico ScorBot ER-4U utilizando comandos seleccionados, utilizando Python para el desarrollado del control e interfaz y Matlab para la simulación de los datos esperados

4.4. Implementación de la solución

Inicialmente se realizó una investigación sobre los tipos de control que se desarrollan en el robot Scorbot ER, en específico el control por matrices cinemáticas, para realizar este tipo de control se necesitó primeramente revisar las condiciones actuales de los sensores y encoders para obtener un sistema sensorial efectivo.

Al iniciar la etapa de revisión, se realizó la lubricación y restauración de las tensiones correctas en cada una de las cadenas para que el sistema mecánico estuviese en correcto estado para la puesta en marcha del proyecto además de la corrección de las posiciones para mejorar el agarre de la cadena además se procedió a realizar una revisión general del cableado para la correcta obtención de las variables a controlar.

Se realizó un ajuste en los motores del codo y hombro, los mismos para disminuir la inexactitud de los movimientos deseados posteriormente, estos ajustes se realizaron varias veces dado que no quedaron lo suficientemente fuertes y otros no fueron acertados, se concluyó que estos motores(codo y hombro) poseen tornillos con desgastes lo que podría ocacionar cada cierto tiempo la necesidad de ajustar nuevamente los mismos antes de llevar a cabo un proceso para no producir una variación importante en la colocación final.

Adicionalmente se llevó a cabo la restauración de dos de los seis circuitos de sensado, para dar con ellos se realizó una prueba de validación de señal infrarojo (se revisó que cuando no estuviese interrumpido el valor de salida fuese HIGH y en caso contrario LOW como se muestra en la figura 4.2) donde la prueba permitió medir los circuitos finales construidos con los sensores y su correcto funcionamiento. Al realizar una revisión general del sistema, se partió a la programación de un módulo para la lectura de los sensores, este consistió en realizar la conexión correspondiente de los sensores a la raspberry pi, la cual permite la adquisicion de datos de manera binaria , la misma permitió dar con 4 sensores que estaban en mal funcionamiento, este procedimiento se iteró varias veces hasta conseguir el funcionamiento adecuado para las banderas de posición HOME, siguiendo el uso de los GPIO de una Raspberry Pi 3 como se muestra

en la figura 4.1.

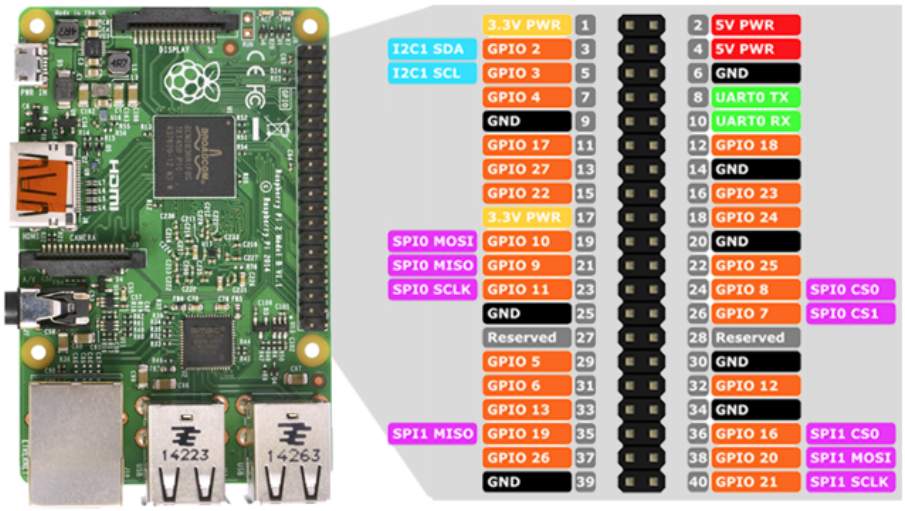


Figura 4.1: Función de los pines de la raspberry pi 3 B+.
Fuente: [41]

Tambien se identificaron mediante estas conexiones que los pines correspondientes al GPIO 23, 24,5,17,4 no son funcionales ya que en ciertos momentos no cumplen los rangos adecuados ni tampoco se pudieron usar para generar salidas al sistema mediante oscilogramas.

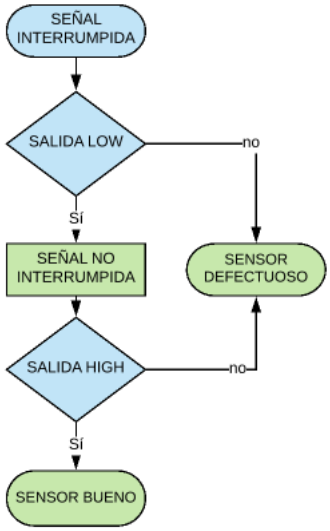


Figura 4.2: Prueba lógica de validación para circuitos infrarojo ITR8102.
Fuente: Elaboración propia

Se realizó la revisión del estado de cada uno de los encoders(mediante la prueba lógica de la figura 4.3) y motores correspondientes, se encontró que uno de los encoders de los motores iniciales fue sustituido por uno comercial KY040, se realizó la programación debida para obtener el funcionamiento de los codificadores y las escalas correspondientes a los angulos según el movimiento del motor una cantidad de ciclos o revoluciones.

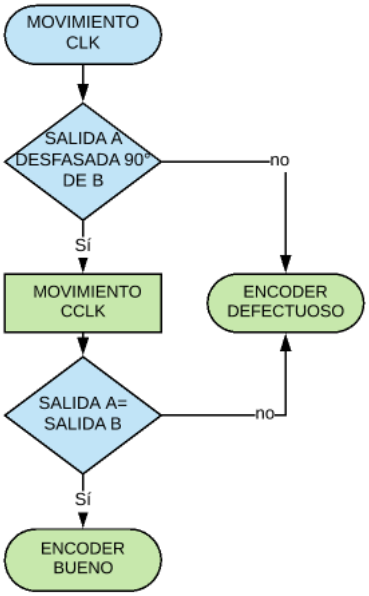
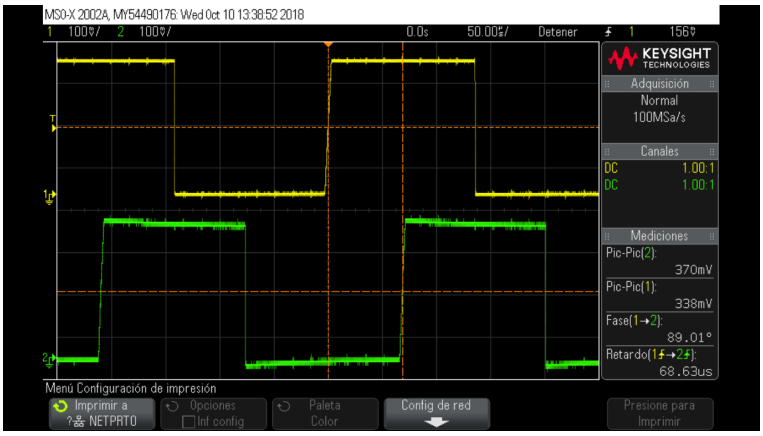
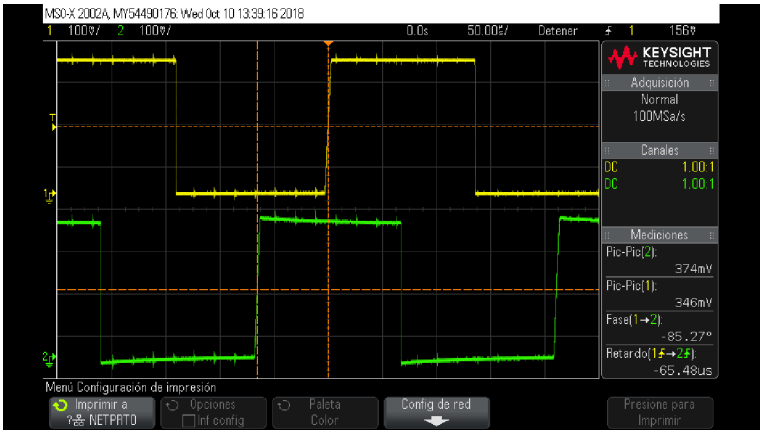


Figura 4.3: Prueba lógica de validación para encoders.
Fuente: Elaboración propia

Seguidamente se realizó la programación para la lectura correcta de los encoders en cada uno de los motores, esto con el fin de tener el conteo correcto del ángulo al que se desplaza el robot mediante pruebas de visualización de las dos salidas A y B en cada encoder(ver a correspondiente a las salidas en movimiento en sentido de las manecillas del reloj y b en movimiento en sentido contrario de las manecillas del reloj, en la figura 4.4) y de repetición para ubicación en el mismo punto y obtención de la escala de los ángulos por cada pulso del encoder. Fue necesario implementar una pieza adicional para fijar el encoder de la base deslizante, ya que las mediciones de este encoder estaban incorrectas al realizar movimientos horizontales.



(a) Movimiento en el sentido horario



(b) Movimiento en el sentido horario

Figura 4.4: Resultado de las pruebas de los encoders obtenidos.
Fuente: Elaboración propia

Se debió realizar un cableado nuevo ya que el anterior presentaba problemas de potencia y perdidas de datos y se realizó el montaje nuevamente de las conexiones con el GPIO, esto para mejorar la calidad de los datos, el control y la estética del proyecto. Se consideró necesario ajustar el cableado de encoders y sensores junto con los cables de los motores de tal manera que permitieran el movimiento libre del brazo, ya que los mismos podrían provocar mal funcionamiento si se da la desconexión por los movimientos que realiza el robot.

Una vez que el sistema sensorial(sensores y encoders) se encontraba en buenas condiciones de medición, se continuó con la programación de las matrices cinemáticas para la posterior implementación y la codificación de las señales

proporcionadas por los encoders para la toma y simulación de datos precisa en Matlab, pero se reformuló el uso de Matlab como solución de interfaz dado que no era compatible con el sistema operativo Raspbian por lo que las simulaciones son meramente teóricas.

De manera paralela a la programación de matrices cinemáticas, se logró establecer la conexión remota entre distintos dispositivos(Raspberry Pi y computador personal) mediante la implementación de la plataforma VNC que permite la obtención de datos en tiempo real y el control remoto del robot, se comprobó que se pueda realizar el intercambio de documentos de texto para su posterior ejecución.

Seguidamente, se realizó el movimiento de los motores mediante programación Python, donde fue necesaria la implementación de un expansor de pines para aumentar la cantidad de variables sensadas y controladas con el fin de obtener el movimiento requerido donde se implementó adicionalmente el sensado con el fin de corroborar el buen funcionamiento del mismo y la implementación del movimiento angular controlado.

Otra etapa adicional fue desarrollar una interfaz gráfica que permitiera la programación del brazo robótico asi como su simulación, por lo que realizaron algunas pruebas para una interfaz inicial en Python pero dado que la misma no presenta los resultados requeridos se elaboró otra en Matlab, seguidamente se elaboró una simulación del espacio de trabajo con los datos que se pueden obtener mediante Matlab y se elaboró un programa(interfaz de usuario) para la programación del robot mediante Processing, este se desarrolló en el modo de programación Java, utilizando botones con funciones de programación más rápida basado en Melfa Basic IV y miscelaneos(abrir, guardar, ejecutar, parar) como se observa en la figura 3.8(e).

Se realizó una reevaluación del tipo de interfaz, ya que se consideró mejorar visualmente y compatibilizar la misma con el controlador implementado asimismo sustitu motor paso pasoir la capacidad del Matlab para cálculos matemáticos con Python y las librerías correspondientes dado que la interfaz inicial es la que se muestra a continuación en la figura 4.5.

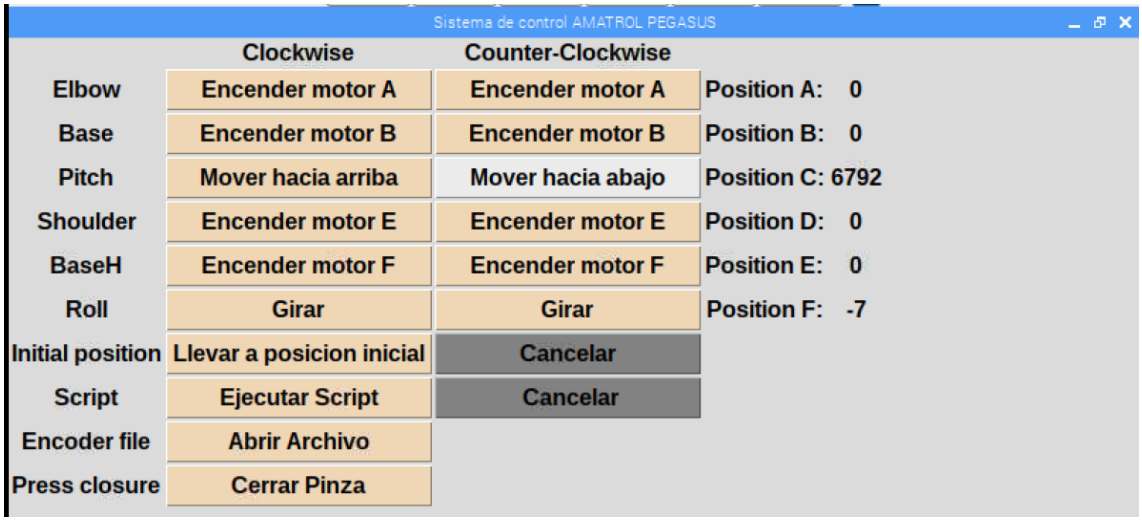


Figura 4.5: Interfaz del robot antes de realizar el proyecto.
Fuente: Elaboración propia

Al poner en marcha la primera solución se tuvieron en cuenta varios aspectos entre ellos que el sistema final diera los sistemas resultantes de tal manera que presentarán en conjunto una robustez para el sistema de control de allí que al realizar la implementación de la comunicación en VNC se encontraron varias ventajas ya que permitió conectar los diferentes modos de control ya sea programado o directo aunque compromete los recursos del sistema. Por otro lado al implementar la interfaz de python tkinter se obtuvo de manera satisfactoria una interfaz que permite el control, visualización pero que no presenta la complejidad requerida ni la estética por lo que se decidió cambiar para tener una interfaz más robusta y dinámica tambien.

Respecto al control del sistema se planteó el uso de sensores, motores y encoders para aplicar la cinemática inversa y directa del robot el mismo no fue replanteado ya que presentaba bastante robustez pero se presentaron varias dificultades dado que al realizar las pruebas de los sensores se encontraron dos de estos en mal estado lo que llevó a la compra de los mismos y el montaje de los circuitos posterior para aplicar las pruebas. En estas pruebas se debió realizar la prueba lógica tanto antes de realizar la compra para su sustitución como después de la misma al realizar el montaje lo que comprometió tiempo. Por otro lado al realizar la implementación de los motores se realizó una prueba lógica que permitió obtener la forma de movimiento de los motores al aplicar voltaje

en positivo y en negativo respecto a las terminales del Driver haciendo de este una forma de encontrar las limitaciones mecánicas de los motores al encontrar estas limitaciones se obtuvieron problemas dado que en algunos de los casos esta limitación debía ser menor por lo que el espacio de trabajo se veía disminuido por los primeros grados de libertad.

En otro sentido al realizar las pruebas lógicas de medición de los 5 encoders se encontró que la prueba no era pasada por cuatro de los seis encoders pero luego mediante pruebas de voltaje con un oscilograma conforme a la prueba que se muestra en la figura 4.3, se encontró que los encoders se encontraban en buen estado dado que todos respondían corrrectamente al estímulo de movimiento suministrado por lo que esto llevó a la conclusión de que al no contemplar en la prueba que podían fallar los pines de la Raspberry Pi, se tuvo que realizar una prueba extra para revisarlos de tal manera que se pudiera conocer cuáles estaban disponibles para el uso y cuáles no podían ser utilizados.

Seguidamente se realizó una prueba para el movimiento de los motores y pruebas mecánicas con los encoders, conociendo de los mismos el movimiento en contrarreloj o en el sentido de las manecillas del reloj en éste se encontraron problemas con la lectura de los dos valores de la salida de los encoders al tener un problema en la lectura de los encoders se intentó resolver de tres maneras distintas y las mismas pruebas se vieron afectadas por el uso de prints(impresiones).

Respecto al sistema mecánico, la solución planteada fue la del mejoramiento mecánico, el cual se llevó a cabo tal como se plantea el mismo se encontraron algunos problemas que son difíciles de resolver dado que ya el robot presenta desgastes en los engranes y con ello en los motores y al no ser un sistema nuevo no permite que estos ajustes se mantengan durante un tiempo deseado lo que conlleva a estar realizando ajustes manuales cada vez que se necesite realizar el uso del robot para disminuir las inexactitudes que pudieran ser introducidas al mover los motores.

Respecto al sistema eléctrico se había elegido el uso del cableado original en el cual resultó ser útil hasta que se realizaron pruebas conjuntas de los sensores encoders y motores en ella se encontró que al realizar un movimiento con los motores los cables se desprenden fácilmente o así en contacto con otros por lo

que se procedió a realizar un rediseño del sistema eléctrico desde el punto de vista del cableado.

Al realizar la programación para control en el Home o la manera de encontrar cómo las inexactitudes mecánicas no influyeran en la ubicación del mismo se realizó una programación que realizará la colocación hacia la derecha y otro que le realizaran el sentido inverso. De manera análoga al realizar la programación de la cinemática directa se tomó en cuenta que la interfaz actual permitía aplicarla de manera lineal pero esto influye en el tiempo ejecución del programa y ralentiza el tiempo de respuesta general del sistema.

4.5. Reevaluación y diseño

En la programación planteada para llegar al punto cero(HOME) se encontró un problema dado que la solución no era pertinente y duraba mucho en la ubicación del punto cero y se consideró que no era óptima, se procedió a buscar un punto estratégico que permitiera llegar a HOME y la desactivación del motor por lo que se pusieron pequeños stickers de color verde para la señalización del punto estratégico. Se piensa que una implementación de un sistema de visión podría mejorar la calidad del sensado y dar una retroalimentación para la ubicación del HOME.

Para obtener la cinemática directa se encontró una manera de no utilizar directamente las matrices sino el resultado aproximado de los valores requeridos a partir de una programación en Matlab simbólico obteniendo X, Y y Z a partir de los valores de los ángulos actuales o requeridos del robot con el fin de disminuir el costo computacional.

Respecto a la solución de la cinemática inversa se tuvieron varias dificultades inicialmente se implementó el uso de matrices inversas para obtener la cinemática inversa y los ángulos correspondientes para llegar al punto requerido pero la implementación de estas matrices simbólicas aumentó la complejidad con respecto a la cantidad de soluciones de los ángulos y esto aumentaba al tiempo necesario de procesamiento de la raspberry así como el uso de recursos.

Se buscó una manera paralela para atacar este problema dónde se encontró una librería que permitía el cálculo de las primeras articulaciones y finalmente las últimas tres eran situadas de acuerdo con los ángulos faltantes para la orientación lo cual no era factible dado que los ángulos finales obtenían valores que están fuera de los límites dando como resultado un punto que no se accedía diferente del requerido con ello se procedió buscar otra manera para obtener la solución requerida.

De este punto se implementaron ecuaciones planteadas en un paper, las cuales no fueron útiles ya que los resultados al ser implementados no daban el punto final al aplicar la cinemática directa, de la misma manera se realizó el uso de una librería que realiza el despeje de las ecuaciones y los ordena en un archivo látex

en PDF pero las soluciones son 2^n de acuerdo con la cantidad de soluciones que se obtienen de las primeras articulaciones dando como resultado 2 soluciones para teta1, 4 soluciones para teta2 8 soluciones para teta 3, 16 soluciones para teta 4 y finalmente 32 soluciones para teta 5 dando como resultado una solución poco eficaz.

Con relación a la solución del movimiento lineal se realizó la parametrización mediante 1000 puntos de cálculo donde se realizó la resta del valor final menos el valor inicial y con esto obtener 1000 puntos después de esto se planteaba realizar la cinemática inversa de cada uno de estos puntos para realizar un movimiento lineal pero se encontró que dado que la cinemática directa fue resuelta y la inversa no pudo implementar por lo que se buscó la manera de crear una programación lineal de los ángulos siendo éstos últimos los calculados mediante la cinemática inversa de ser esta última correcta.

Por otra parte la implementación de la interfaz y que de esta no se obtuvieran los resultados deseados, se realizó una segunda interfaz con la extensión ".fig.$^{\text{en}}$ Matlab pero tampoco fue funcional dado que la Raspberry no posee compatibilidad para la ejecución de este tipo de paquetes seguidamente se planteó el uso de una extensión ".mApp"de Matlab tambien se encontró que a pesar de que la aplicación es bastante robusta presentó dificultades como la comunicación con el control actual por lo que entonces se replanteo el control para el mismo dando como resultado una aplicación que permitía un control de los motores y lectura de los sensores pero al tomar tanto tiempo para la ejecución se descartó la posibilidad de implementarlo como solución definitiva y se utilizó una tercera propuesta con el programa Processing mismo que fue desarrollado con botones, en éste se logró encontrar una solución más optimizada dado que la cantidad de botones permitía reducir el espacio de programación e incluir en la misma ventana un control directo.

Al realizar lo planteado anteriormente se obtuvo una interfaz en Processing y se realizó una investigación sobre los tipos de protocolo o formas de conexión entre una interfaz y un sistema de control se utilizaron varias soluciones entre ellas el uso de documentos de texto, documentos csv además de una plataforma maestro-esclavo donde la interfaz se comunica con el control siendo la primera un cliente y el control un servidor.

Para llegar a tal conclusión se realizó una prueba en cada uno de los sistemas propuestos como se muestran en la figura 4.6, en las cuales se puede observar que la calidad del programa no es la deseada, que se dan problemas al intentar ejecutar programas que se observan en las figuras 4.7 y 4.8 de tipo .fig o .mApp, dado que el sistema raspbian no posee compatibilidad directa y tampoco permite un interfazado inmediato entre Matlab y Raspbian y en caso de ser posible se plantea el empleo de documentos csv para el seguimiento de las variables y peticiones del usuario al servidor o consultas de datos.

Dado que no se tomó en cuenta la compatibilidad de matlab se realizaron varias interfaces de prueba en matlab que resultaron poco exitosas al no poder ejecutarse y necesitar un protocolo de comunicación con el control. Del mismo modo se desarrolló una interfaz en Processing correspondiente a la primera iteración(ver figura 4.9) que dio resultados de robustez y esteticamente cumplió los requisitos para seguidamente ser mejorado en las iteraciones siguientes.

Figura 4.6: Interfaz elaborada con Python TKinter.
Fuente: Elaboración propia

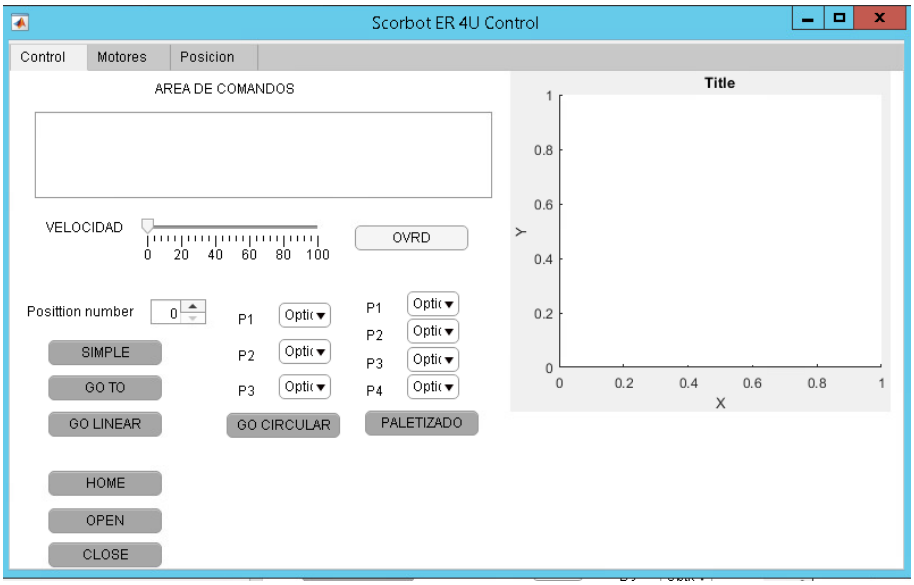


Figura 4.7: Interfaz elaborada con Matlab ".*mApp*".
Fuente: Elaboración propia

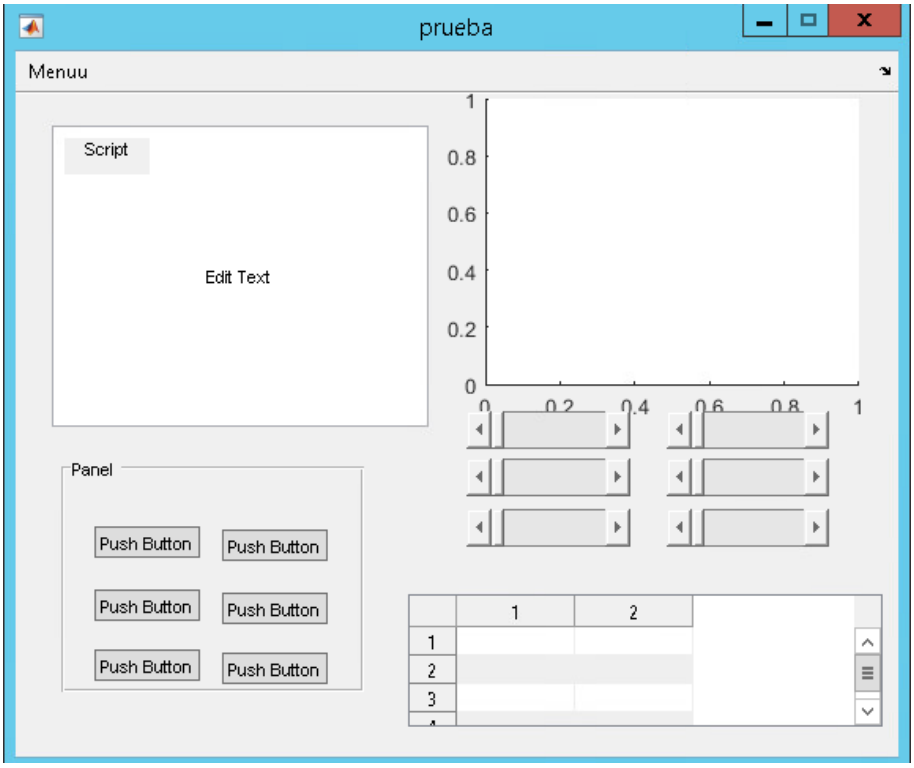


Figura 4.8: Interfaz elaborada con Matlab ".*fig*".
Fuente: Elaboración propia

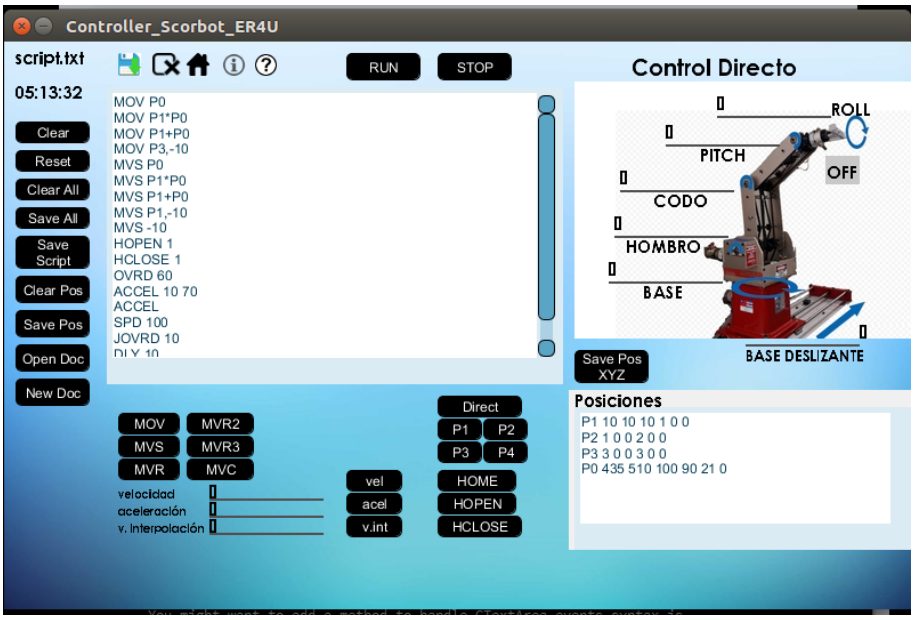


Figura 4.9: Primera interfaz elaborada con Processing ”.*pde*”.
Fuente: Elaboración propia

Capítulo 5

Descripción detallada de la solución

En este capítulo se explica el diseño de la solución implementada, en este se hace un análisis de las soluciones y la selección final de cada una de ellas, la descripción del hardware y del software.

5.1. Análisis de soluciones y selección final

Tabla 5.1: Iteraciones dadas para encontrar la solución final.

Sistema	Propuestas	s1	s2	s3	s4	s5	s6	s7	s8
Control	Remoto Rasp. Local Rasp. Remoto Arduino	X	X	X	X	X	X	X	X
Potencia	Batería Fuente Poder Panel solar	X	X	X	X	X	X	X	X
Comunicación Remota	ssh Amazon VNC	X	X	X	X	X	X	X	X
Controlador	Raspberry PLC Arduino	X	X	X	X	X	X	X	X
Interfaz	Matlab Python Processing	X	X	X	X	X	X	X	X

Fuente: Elaboración propia

Nota: la "X"denota la solución implementada, cuando se encuentra en rojo fue donde se decidió darle continuidad para obtener la solución final

Como se puede notar en la tabla 5.1,la cantidad de iteraciones realizadas para llegar a la solución final pueden resumirse en ocho, dado que son las más relevantes porque contemplan cambios circunstanciales introducidos en el desarrollo de la solución que divergen de la solución anterior de manera considerable.

En la primera de ellas se puede ver que se escogió un sistema de tipo local para la conexión de la Raspberry y su mando seguidamente se seleccionó la fuente de poder como solución, además del ssh como comunicación remora y un controlador de tipo Raspberry Pi 3 B+ y como interfaz python de allí se encontraron varias varios inconvenientes de los cuales se logró cambiar la interfaz.

La segunda iteración fue para generar una interfaz acorde, en este caso con la segunda propuesta que era matlab en ella se desarrollaron dos aplicaciones

las mismas son de tipo ejecutable pero ninguna de las dos logró dar la solución requerida por lo que se volvió a hacer una interacción que corresponde a la solución 3 donde se encuentra que se tomó como la solución más viable y factible.

Posteriormente se realizó una prueba con adquisición de datos mediante Arduino Uno, la misma no fue factible dado que no existía la cantidad de pines de entrada y salida necesarios para la conexión de los sensores y los encoders más la colección de los motores y la pinza por lo que se procedió a hacer una cuarta solución en la que se cambió el tipo de conexión con el sistema remoto porque no permitía visualización de la interfaz desarrollada y en ese se intentó realizar una conexión mediante la plataforma de Amazon Services ioT la misma no fue funcional dado que una limitación del proyecto es el desarrollo de un control de bajo costo y para el uso del mismo se necesitaba realizar un pago anual por lo que no fue una solución viable de allí sale la solución cinco(s5), la cual es la implementación de la plataforma VNC, la misma permitió realizar un control remoto eficaz donde se puede controlar el controlador específicamente todo su sistema y además de eso en la interfaz que se creó se puede realizar la programación y ejecución externa mediante terminal para la instalación de los paquetes necesarios o conexion a internet.

Más tarde se efectuó una revisión del sistema de potencia donde se planteó el uso de una batería externa o una fuente externa pero dado que la batería no daba la suficiente duplicidad y multifuncionalidad de voltajes se prefirió seguir usando la fuente de poder. De manera análoga la solución 7 corresponde al cambio de un control de local a remoto, se debió implementar mediante el uso de servidores y clientes para el manejo y adquisición de los datos pero no fue lo suficientemente eficiente dado que se debían enviar los archivos que se creaban y eso podía ocasionar la perdida de datos y provocar que el sistema no pudiese verificarlos por lo que la solución eficaz final seleccionada fue el sistema local de mando mediante la raspberry pi y en conjunto con una fuente de poder, una conexion VNC para el control remoto de los usuarios mediante la implementación de una raspberry y un sistema de interfaz en Processing desarrollado en lenguaje de Java.

5.2. Descripción del hardware

La solución final que se ha implementado para el proyecto consta de cinco sistemas como se muestra en la figura 5.1 con una conexión mostrada en la Tabla 5.2 .

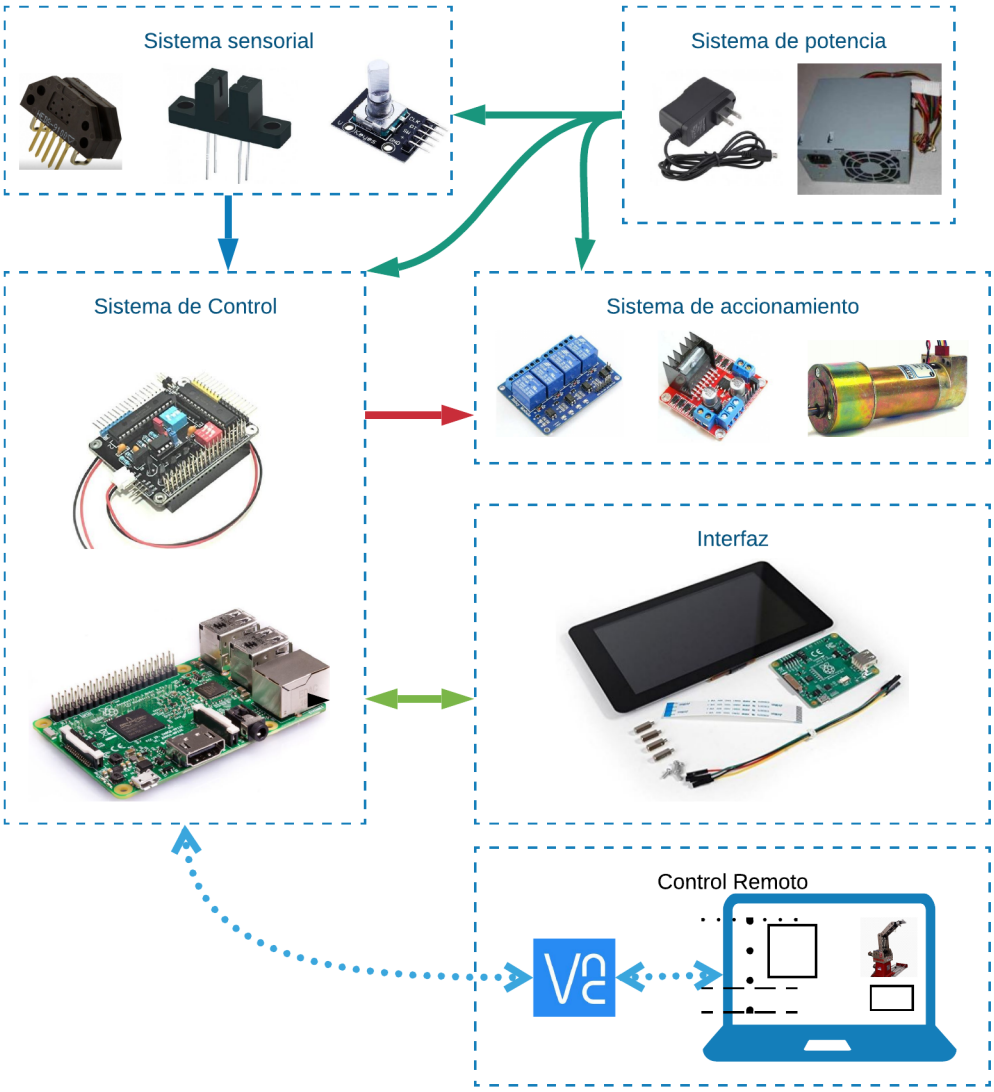


Figura 5.1: Representación del hardware implementado en la solución.
Fuente: Elaboración propia en draw.io

Tabla 5.2: Conexión de las variables en el GPIO del controlador.

Pin	Señal	Designación	Función
12	Sensor codo	input	low activa posicion 0 del codo
20	Sensor base	input	low activa posicion 0 de la base
7	Sensor pitch	input	low activa posicion 0 del pitch
16	Sensor roll	input	low activa posicion 0 del roll
21	Sensor hombro	input	low activa posicion 0 del hombro
27	Sensor final de carrera	input	low activa posicion 0 del final de carrera
25	Encoder motor A salida A	input	flanco de subida activa aumento o disminución de ciclos del motor A
8	Encoder motor B salida A	input	flanco de subida activa aumento o disminución de ciclos del motor B
15	Encoder motor C salida A	input	flanco de subida activa aumento o disminución de ciclos del motor C
18	Encoder motor D salida A	input	flanco de subida activa aumento o disminución de ciclos del motor D
14	Encoder motor E salida A	input	flanco de subida activa aumento o disminución de ciclos del motor E
10	Encoder motor F salida A	input	flanco de subida activa aumento o disminución de ciclos del motor F
19	Encoder motor A salida B	input	flanco de subida activa aumento o disminución de ciclos del motor A
9	Encoder motor B salida B	input	flanco de subida activa aumento o disminución de ciclos del motor B
11	Encoder motor C salida B	input	flanco de subida activa aumento o disminución de ciclos del motor C
13	Encoder motor D salida B	input	flanco de subida activa aumento o disminución de ciclos del motor D
6	Encoder motor E salida B	input	flanco de subida activa aumento o disminución de ciclos del motor E
22	Encoder motor F salida B	input	flanco de subida activa aumento o disminución de ciclos del motor F
8	Motor A Derecha	output	high activa movimiento a la derecha.
7	Motor A Izquierda	output	high activa movimiento a la izquierda.
6	Motor B Derecha	output	high activa movimiento a la derecha.
5	Motor B Izquierda	output	high activa movimiento a la izquierda.
4	Motor C Derecha	output	high activa movimiento a la derecha.
3	Motor C Izquierda	output	high activa movimiento a la izquierda.
2	Motor D Derecha	output	high activa movimiento a la derecha.
1	Motor D Izquierda	output	high activa movimiento a la izquierda.
9	Motor E Derecha	output	high activa movimiento a la derecha.
10	Motor E Izquierda	output	high activa movimiento a la izquierda.
11	Motor F Derecha	output	high activa movimiento a la derecha.
12	Motor F Izquierda	output	high activa movimiento a la izquierda.
14	Switch Pinza	output	high activa apertura de la pinza.
2	SDA	output	high activa datos I2C
3	SCL	output	high activa clock I2C

Fuente: Elaboración propia

El primer sistema consta del sistema mecánico, al que se le realizó una limpieza y montaje, tensado de cadenas y lubricación respectiva. El segundo sistema fue el de potencia, el cuál se basó en la implementación de una fuente de poder de computadora(ver figura 5.2), que tiene las características que se muestran en la tabla 5.3, la misma fue seleccionada debido a la necesidad de diferentes potenciales y la potencia para entregar y su estabilidad de energia para el control y movimiento de los motores.

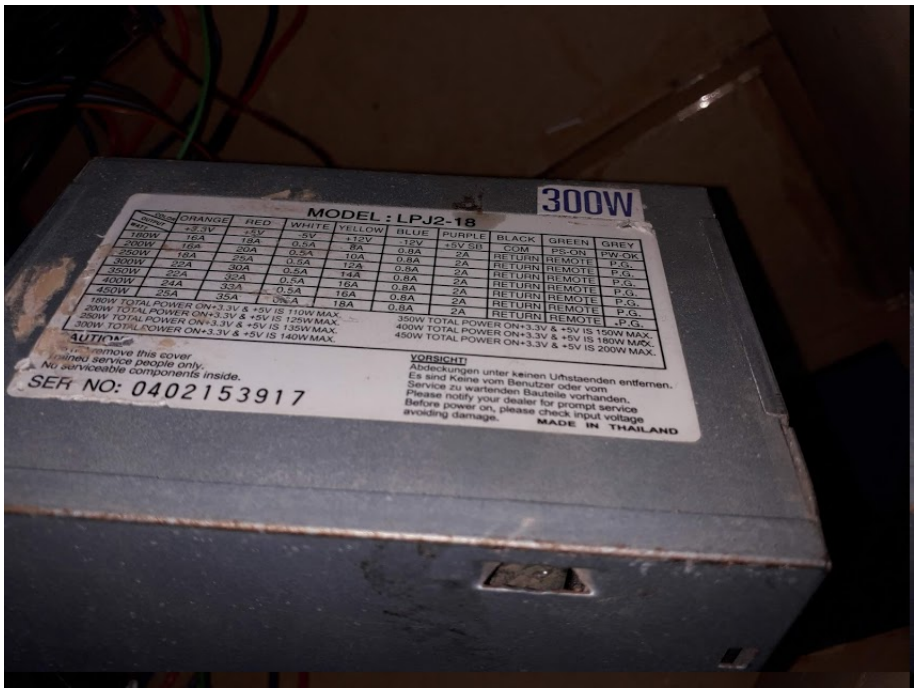


Figura 5.2: Fuente de poder utilizada para dar solución al sistema de potencia.
Fuente: propia

Tabla 5.3: Características del sistema de potencia utilizado.

COLOR	ORANGE	RED	WHITE	YELLOW	BLUE	PURPLE	BLACK	GREEN	GREY
OUTPUT WATT	+3.3V	+5V	-5V	+12V	-12V	+5V SB	COM	PS-ON	PK-OK
180W	16A	18A	0.5A	8A	0.8A	2A	RETURN	REMOTE	PG
200W	16A	20A	0.5A	10A	0.8A	2A	RETURN	REMOTE	PG
250W	18A	25A	0.5A	12A	0.8A	2A	RETURN	REMOTE	PG
300W	22A	30A	0.5A	14A	0.8A	2A	RETURN	REMOTE	PG
350W	22A	32A	0.5A	16A	0.8A	2A	RETURN	REMOTE	PG
400W	24A	32A	0.5A	16A	0.8A	2A	RETURN	REMOTE	PG
450W	25A	35A	0.5A	18A	0.8A	2A	RETURN	REMOTE	PG

Fuente: Elaboración propia

Despues de realizar un estudio de la cantidad de variables a medir(ver tabla 5.2), se escoge una Raspberry Pi 3 como controlador (ver figura 5.3) además de la implementación de un expansor de pines con el fin de habilitar los pines necesarios para el sistema de control. Adicionalmente se interconectó a una pantalla LCD de 7"para la correcta visualización de la interfaz para el usuario local. La Raspberry realiza el monitoreo del posicionamiento de cada una de las articulaciones mediante los sensores y encoders del sistema sensorial.

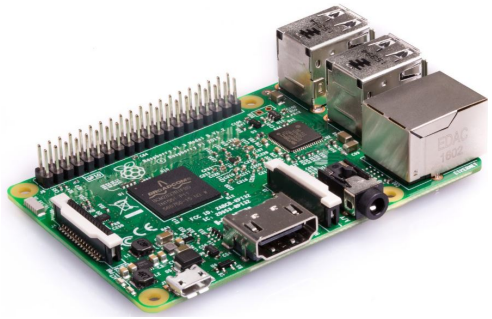


Figura 5.3: Controlador utilizado para la solución.
Fuente: [29]

El tercer sistema constó de la implementación del circuito para la medición de posición con los sensores optoreflectivos para el sensado de la posición HOME o inicial así como la implementación de los encoders debidamente incorporados a los motores correspondientes, los cuales fueron conectados mediante un bus a el expansor de pines que se muestra en la figura 5.4.

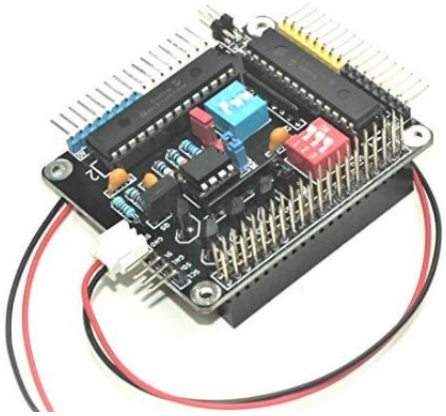


Figura 5.4: Expansor de pines agregado al sistema.
Fuente:[36]

El cuarto sistema consta de la instalación de una pantalla LCD de 7 pulgadas(ver figura 5.5), esta permite la visualización y control local al usuario así como obtener información del controlador.Además de este sistema se consideró un sistema de comunicación para hacer posible el control remoto, por lo que fue necesario utilizar el módulo de ethernet o wi-fi para la conexión de tipo ssh desde un ordenador remoto.

Figura 5.5: Representación de la pantalla LCD utilizada
Fuente: [29]

5.3. Descripción del software

Se realizó una solución con el uso de Raspbian, sistema operativo de Raspberry, donde se hizo uso de Python para la adquisición y almacenamiento de datos así como el control general del sistema. Matlab para la simulacion de los datos obtenidos. En la figuras 5.6, 5.7 y 5.8 se muestra la estructura del software que se siguió para el control general y la solución del mismo en el software, donde se implemento una arquitectura maestro-esclavo para el control efectivo de cada motor según el objetivo general.

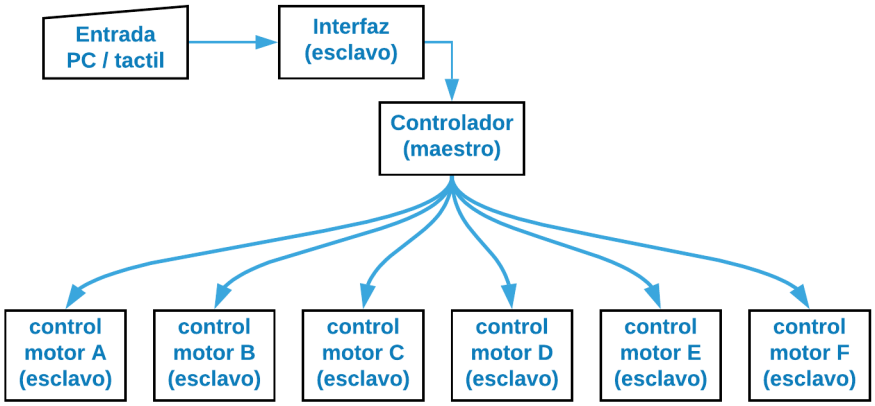


Figura 5.6: Estructura general para el control directo del sistema.
Fuente: Elaboración propia

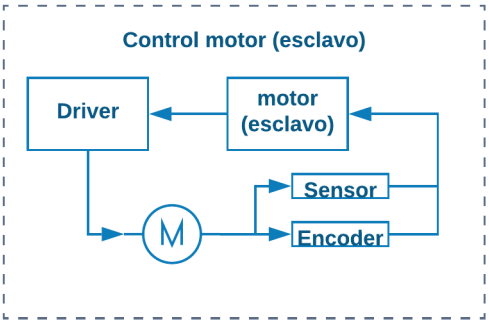


Figura 5.7: Estructura para el control de cada motor.
Fuente: Elaboración propia elaborado en draw.io

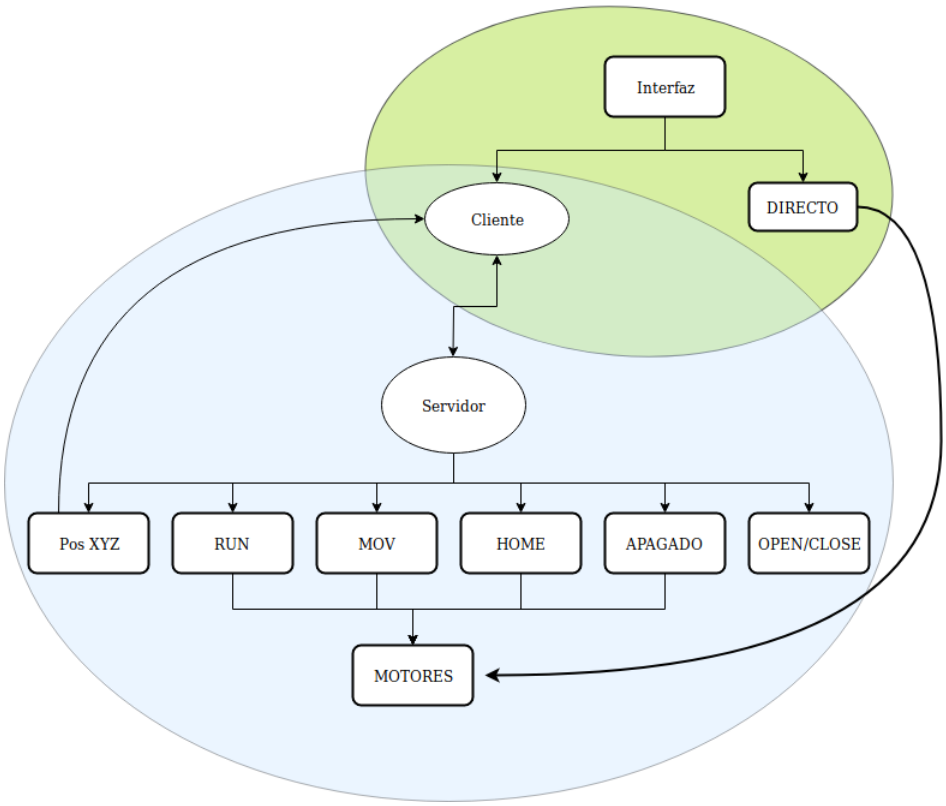


Figura 5.8: Estructura para el control general del sistema robótico.
Fuente: Elaboración propia elaborado en draw.io

Se implementaron distintos módulos para la lectura de los encoders y sensores, se implementaron librerías para el cálculo de las distintas posiciones de las articulaciones, las medidas del sensor permiten llegar a HOME y inicializar en cero el contador de ciclos del motor respectivo. Siguiendo la cinemática del robot, se utilizaron las siguientes longitudes(ver Tabla 5.4) para obtener los posicionamientos respectivos del sistema de referencia XYZ seleccionado[16]. Inicialmente con la figura 5.9 se obtuvieron los parámetros para las matrices que se encuentran en la tabla 5.5 y seguidamete se realizó una modificación para cumplir con la posición de la figura 5.10 y con ello dar con los parámetros mostrados en la tabla 5.6, seguidamente se realizó la multiplicación de todas las matrices de transformación con la finalidad de obtener la cinemática directa e inversa y dar solución al movimiento.

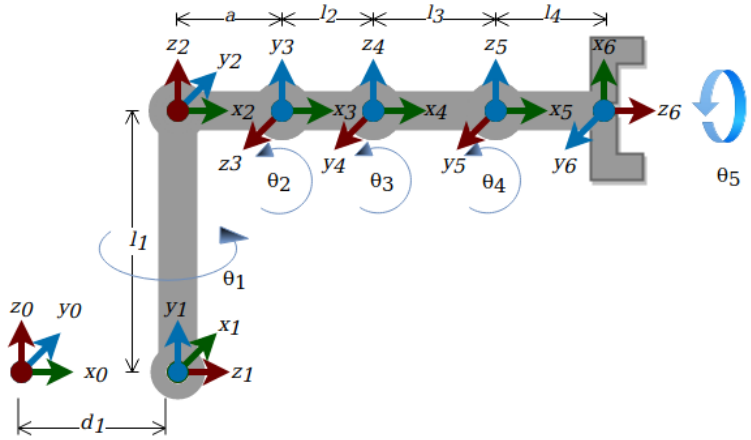


Figura 5.9: Configuración inicial del Scorbot ER-4U para obtener los parámetros de Denavit-Hartenberg.

Fuente: Elaboración propia elaborado en draw.io

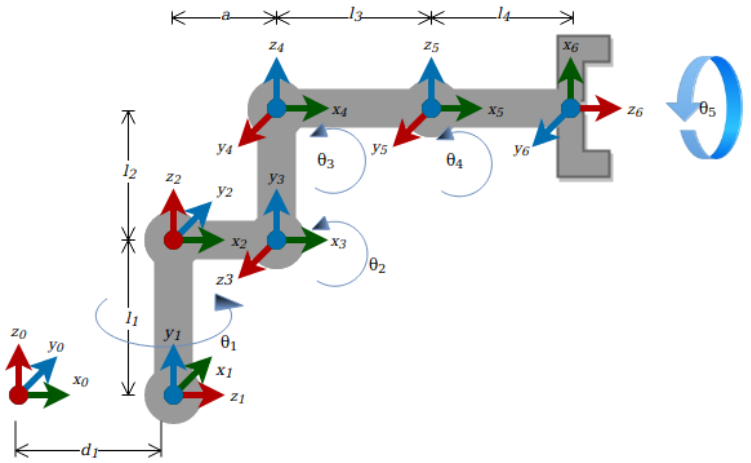


Figura 5.10: Modificación de la articulación e(hombro) para obtener los parámetros de Denavit-Hartenberg desde la posición HOME.

Fuente: Elaboración propia elaborado en draw.io

Tabla 5.4: Longitudes del robot SCORBOR ER 4U a rehabilitar.

articulación	base deslizante(d_1)	base(L_1)	hombro(L_2)	codo(L_3)	pitch+roll(L_4)	a
medida[mm]	1000	410	230	230	140	25

Fuente: Elaboración propia

Tabla 5.5: Parámetros del robot SCORBOR ER 4U para el análisis cinemático de la configuración que se muestra en la figura 5.9.

	alfa	a	theta	d
0-1	$\pi/2$	0	0	-d_1
1-2	-$\pi/2$	0	θ_1-$\pi/2$	L_1
2-3	$\pi/2$	a	θ_2	0
3-4	0	L_2	θ_3	0
4-5	0	L_3	θ_4	0
5-6	$\pi/2$	0	$\pi/2$+ θ_5	L_4

Fuente: Elaboración propia

Tabla 5.6: Parámetros del robot SCORBOR ER 4U modificada al definir la posición HOME.

	alfa	a	theta	d
0-1	$\pi/2$	0	0	-d_1
1-2	-$\pi/2$	0	θ_1	L_1
2-3	$\pi/2$	a	θ_2+$\pi/2$	0
3-4	0	L_2	θ_3- θ_2-$\pi/2$	0
4-5	0	L_3	θ_4- θ_3	0
5-6	$\pi/2$	0	$\pi/2$+ θ_5	L_4

Fuente: Elaboración propia

Se realizó la implementación de los parámetros para obtener el modelo cinemático mediante la implementación de las matrices directas correspondientes tal como se muestra a continuación:

$$A_{i-1}^{i} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & a \\ \cos\alpha \cdot \sin\theta & \cos\alpha \cdot \cos\theta & -\sin\alpha & -d \cdot \sin\alpha \\ \sin\alpha \cdot \sin\theta & \sin\alpha \cdot \cos\theta & \cos\alpha & d \cdot \cos\alpha \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.1}$$

La ecuación 5.2 detalla la forma de obtener mediante matrices cinemáticas la posición y orientación mediante la multiplicación de las matrices simples obtenidas realizando el uso de los parámetros del robot y los ángulos actuales o requeridos para conocer el punto final en términos de X,Y,Z.

$$A_0{}^6 = A_0{}^1 \cdot A_1{}^2 \cdot A_2{}^3 \cdot A_3{}^4 \cdot A_4{}^5 \cdot A_5{}^6 = T = \begin{bmatrix} n_x & o_x & a_x & X \\ n_y & o_y & a_y & Y \\ n_z & o_z & a_z & Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

Se realizó una aproximación de los puntos de movimiento X, Y, Z utilizando el método geométrico para el análisis de cinemática del robot y se obtuvieron las ecuaciones que simplifican los calculos de tal manera que se pueda minimizar el tiempo de análisis de datos para mejorar el control como se muestra en las ecuaciones (5.3),(5.4),(5.5).

$$X = (l_3 \cdot \cos(\theta_3) - l_2 \cdot \sin(\theta_2) + l_4 \cdot \cos(\theta_4) + a) * cos(\theta_1)) \tag{5.3}$$

$$Y = (l_3 \cdot \cos(\theta_3) - l_2 \cdot \sin(\theta_2) + l_4 \cdot \cos(\theta_4) + a) \cdot \sin(\theta_1) + d1 \tag{5.4}$$

$$Z = l_3 \cdot sin(\theta_3) + l_2 \cdot \cos(\theta_2) + l_4 \cdot \sin(\theta_4) + l_1 \tag{5.5}$$

Capítulo 6

Análisis de Resultados

6.1. Resultados

Primeramente, se logró diseñar satisfactoriamente un sistema de control de posicion y desplazamiento para el brazo robótico Scorbot ER-4U ubicado en el Instituto Tecnológico de Costa Rica no así su completa implementación.

Adicionalmente se restauró mecánicamente mediante la lubricación, tensado de las cadenas así como ajustar los ejes con una mayor tensión y eléctricamente mediante la implementación de un cableado nuevo, el uso de pines macho de 0.25mm para el acople con una tira de 2x20 como se muestra en la siguiente figura(ver figura 6.1), con esto último se mejoró el sistema de realimentacion de posición de manera que se puedan ejecutar desplazamientos precisos en cada uno de sus ejes.

Figura 6.1: Bus de datos utilizada para el acople de los datos de sensado, encoders, I2C y potencia con el módulo del expansor de pines HAT32IOPE.
Fuente: [29]

Se desarrolló satisfactoriamente el diseño e implementación de una interfaz en Processing, la misma permite realizar una manipulación directa de las secuencias programadas asímismo realizar distintas peticiones al sistema de control, como resultado, se obtuvo la siguiente interfaz(ver figura 6.2 la cuál posee listas desplegables con funciones miscelaneas para la apertura de documentos o creación de uno nuevo, asímismo proporciona un control directo mediante deslizadores y un control programado, botones de programación rápida con los comandos Melfa Basic 4 seleccionados para el control, en los que se deben seleccionar las posiciones a utilizar como posición final, posiciones de transito para la inserción en el script de comandos.

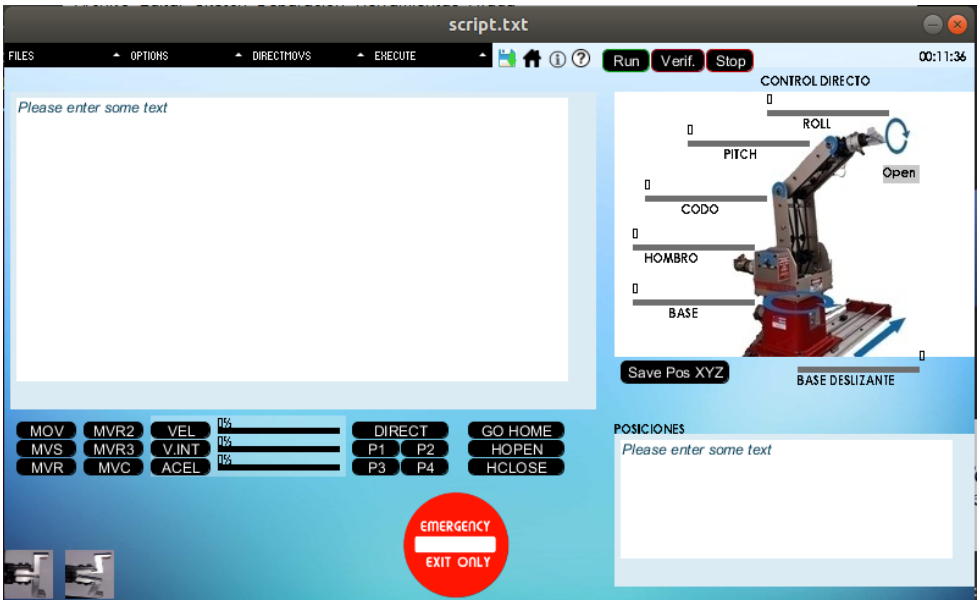
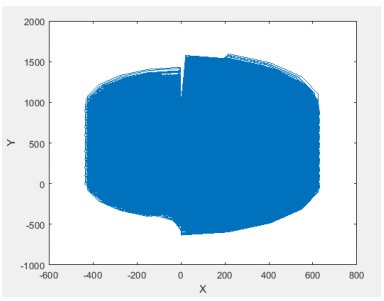
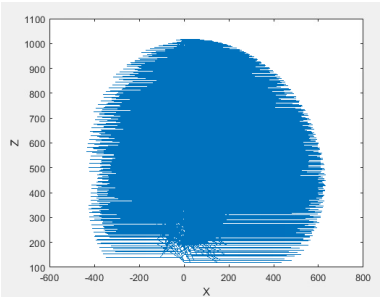


Figura 6.2: Interfaz generada para el control del brazo robótico con matrices cinemáticas.
Fuente: Elaboración propia

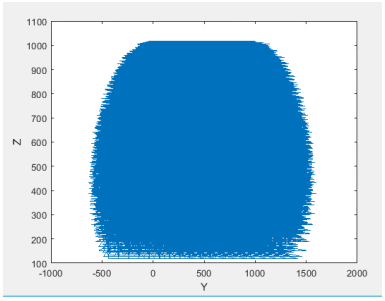
Se consiguió positivamente el establecimiento de un enlace punto a punto entre el brazo robótico ScorBot ER-4U y el usuario remoto mediante el uso de la plataforma VNC, la cual hace uso de una conexión por proxy en un grupo de dispositivos específicados en el usuario para controlar el sistema de manera remota, el mismo realiza un control completo, desde el cambio de parámetros del sistema operativo del mismo, acciones programadas de actualización e instalación, transferencia de archivos y mensajes para el control guiado del mismo y precisamente en la interfaz permite una vista completa, control programado y toma de decisiones del usuario respecto al controlador Raspberry Pi 3 B+. Se logró obtener el espacio de trabajo mediante la simulación de las ecuaciones aproximadas de X,Y,Z dando como resultado los posibles movimientos de control que se pueden obtener mediante la implementación de este control cinemático(ver figura 6.3).



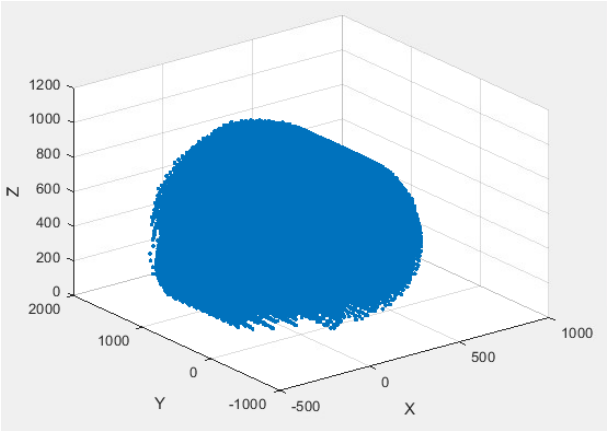
(a) Espacio de trabajo XY del Scorbot ER-4U



(b) Espacio de trabajo XZ del Scorbot ER-4U



(c) Espacio de trabajo YZ del Scorbot ER-4U



(d) Espacio de trabajo XYZ del Scorbot ER-4U

Figura 6.3: Representación del espacio de trabajo del Scorbot ER-4U mediante la simulación en Matlab.

Fuente: Elaboración propia a partir de la implementación de las matrices cinemáticas obtenidas

6.2. Análisis

Primeramentee, se obtuvieron los programas con python para la lectura de los sensores que permiten el cambio de cuatro de ellos de manera satisfactoria. Seguidamente los encoders fueron revisados para entender su funcionamiento e implementación del mismo en el control cinemático, y fue la solución de sensado la que se optimizó de tal manera que la lectura de los mismos solo se realizara al realizar un movimiento con el hilo correspondiente.

Paralelamente se realizó una restauración del brazo robòtico de manera que permitiera un movimiento sin cabeceos, para ello se realizaron un ajuste de ten-

siones de las cadenas y un ajuste de altura del motor A, dado que este motor posee un problema de desgaste, el torque generado por el movimiento cambia el ajuste de manera casi automática al realizar un movimiento con este por lo que se recomienda cambiar la pieza o darle una solución de manera correctiva, ya que no se encontró la manera adecuada para disminuir totalmente los cabeceos provocados y que su estado se mantuviese así.

Al realizar la programaciòn mediante Processing en el lenguaje de Java, se obtuvo la interfaz que se muestra en la figura 6.2, en la misma implemento una estructura de esclavo, para enviar los datos recibidos por el usuario y es enviado al controlador para la activación de los motores como si este último jugara el papel del maestro orquestando todo el control, en este proceso de diseño no se tomó en cuenta que la creación de una interfaz puede implicar aplicar soluciones para la transmisión de información entre aplicaciones por lo que al no haberse seleccionado inicialmente, se dedicó tiempo en darle una solución que no fue óptima varias veces y dado a ello, el tiempo para dar la solución al problema tambien se vio reducido y aunado a esto el tiempo para aprender y aplicar un lenguaje tambien influencia que el control cinemático no se llevara a su correcta y completa implementación.

Se hizo uso del programa VNC para realizar la conexión de un usuario remoto exitosamente, con el fin de obtener un control y programación del mismo remotamente donde el 100 porciento de los comandos eran digitados y verificados correctamente antes de realizar una ejecución, para llegar a esta solución se buscó inicialmente la propuesta que no permitía compartir las acciones ni pantalla por lo que se buscó utilizar la tecnología de los servicios de Amazon que además de resultar útil podría disminuir el tiempo de recuperación de datos no fue posible dado que el presupuesto es limitado se pretende dar una solución a largo plazo.

Al realizar la programación del control del sistema de control en el lenguaje Python, se realizaron varias iteraciones con las programaciones entre ellas entre ellas encontró la más factible conforme se reiterada por lo que sólo se tomarán en cuenta aquellas limitaciones y problemas que se consideren más importantes por ejemplo inicialmente al tener la solución de la interfaz en python no se pensó en un protocolo para la comunicación entre la interfaz y el control del mismo por

lo que cuando se tenía la interfaz no se lograron transmitir datos entre los mismos y con ello no había un control eficiente ulteriormente se planteó el uso de un servidor y un cliente de manera que la interfaz fuera el servidor y python fuese cliente y la primera enviara las consultas necesarias del usuario para saber cuál era el estado actual del mismo mediante una arquitectura maestro-esclavo pero se encontró que era más factible si se usaban viceversa en terminos de disminución del código necesario para la conexión y envío, por lo que se implementó finalmente al cliente en processing mediante los siguientes comandos:

```
//declaracion del cliente-esclavo
Client myClient;
//conexion del cliente con el servidor en el puerto 5204 del host
myClient = new Client(this, "127.0.0.1", 5204);
myClient.write("7");
```

Los comandos anteriores permitían la definición, conexión y la comunicación al servidor donde esté finalmente lo recibía para realizar el control requerido. Además de eso se tuvo en cuenta que al realizar los controles de los motores hubieron varias falta de conocimiento por parte del desarrollador entre ellas que al realizar la activación del motor C se debía tener en cuenta que esté solo no permitiría el movimiento de la articulación C y por lo tanto se debió realizar una iteración tomando en cuenta que los motores C y D se encontraban ligados donde para mover la articulación C debía de moverse en sentido contrario las banderas de control del motor c y las del motor D en el mismo sentido, para lograr en conjunto un movimiento hacia abajo y o hacia arriba sin embargo para el posicionamiento y movimiento del motor D debía ser configurado con todas banderas de control iguales. Por lo que el poco conocimiento de mecánica tambien obstruyó y obstaculizó encontrar la programación correcta y que atravez de iteraciones se logró dar una solución aunada para que los movimientos no tuviesen problemas.

Otro de los problemas encontrados fue que al realizar el movimiento de los encoders en conjunto con los motores se tenía una mala interpretación de los datos los mismos no se encontraban como se esperaba en la teoría donde se realizó una programación que permitía mediante interrupciones captar los movimientos del mismo y realizar un conteo para conocer las revoluciones por minuto en que se encontraba y realizar el control de velocidad, lo cual no se logró dado que

al realizar un movimiento tanto en sentido horario como antihorario no sé tenía una suma correcta ni constante de los movimientos. A pesar que el movimiento si era constante por lo que se decidió realizar una programación distinta donde se realizará una interrupción después de que la primera interrupción de la salida se daba pero al ser esté tan lento y el tiempo de captación tan pequeño mediante el cual no permitía la correcta captación y procesamiento de los datos y por el mismo motivo no se logró un conteo general correcto por lo que se implementó finalmente el uso de las entradas de los encoders para revisarlas cada vez que se realizará un movimiento del motor hacia la derecha o hacia la izquierda pero el problema es que si era movido cuando se encontraba apagado, se realizara un HARDHOME para solventar cualquier error que se pudiese ocacionar al apagarse ya sea por el botón de emergencia o por falta de suministro eléctrico.

Con respecto a los sensores se tuvo una problemática mecánica que se logró resolver de una manera simple dado que el sistema no logra captar uno de los sensores de la manera correcta dado que la interrupción era demasiado pequeña por lo que se decidió realizar un envolvimiento para mejorar el ancho de la interrupción aunque con esto se ve disminuida la capacidad del sensor para captar un punto específico y existen variaciones de aproximadamente 5 grados para el motor C.

Cabe rescatar que en esta solución de VNC se realizó una conexion de wi-fi permitiendo la conexión de la raspberry a una red a la red de la universidad la cual no fue satisfactoria dado que realizaba la activación de una bandera que terminaba por eliminar los archivos y corromper el sistema como se muestra en la figura 6.4. Despues de esta corrupción se debían volver a instalar los paquetes y programas por lo que afectó de manera sustancial el avance general del proyecto.Luego a modo solución se realizó una conexión en este caso mediante ethernet(cableado) y el mismo resultó ser la solución más viable para la conexión con internet y que permitiera el control remoto del mismo de manera estable.

Figura 6.4: Error al activar el wifi de la raspbeerry y utilizar una conexión WPA Enterprise con protocolo MSCHAPv2.

Fuente: Elaboración propia

Otro factor que afectó la resolución concreta del proyecto es la falta de planificación dado que no se contempló que el cambio de año deshabilitara el acceso y con este, el avance del proyecto en la etapa de mediciones no alcanzó para la finalización del mismo y dado que el problema por resolver la cinemática inversa no se lograba de manera correcta, no se realizó su implementación pues los valores se encontraban fuera de los rangos de los ángulos seleccionados, sin embargo la lectura de los mismos y las interpolaciones para realizar los movimientos, en caso de que funcionara, estaban en correcto funcionamiento. Se realizó la busqueda de otros metodos para resolver la cinemática inversa con tal de conocer los valores de los ángulos a partir de la posición XYZ, entre ellas se puede rescatar que se encontró una resolución mediante redes neuronales pero dado que la implentación de las mismas en una raspberry pi no es factible se dejó como tentativa para dar una solución en un futuro.

Adicionalmente, en el diseño de la interfaz se desarrolló un modelo en 3D para la simulación de los movimientos actuales o del script pero el mismo no se implementó dado que los movimientos no se podían ejecutar dado que sin la cinemática inversa, no se podían conocer los ángulos futuros y con ello tampoco las matrices para la ubicación actual del brazo, sin embargo la verificación

realizada en la interfaz permitió enviar un documento script para ejecutar que se encontrara correcto por lo que a pesar de no ejecutarlos si realizaba la lectura de los comandos básicos del lenguaje Melfa Basic 4 elegidos para dar control al brazo robótico SCORBOT ER-4U en el Instituto Tecnológico de Costa Rica.

Las matrices cinemáticas directas se lograron resolver y aplicar a la interfaz de tal manera que del control directo se tomaron los valores de los ángulos actuales para ingresar al documento de posiciones y dado que se realizó un análisis para setear los ángulos actuales como HOME, se realizaron varios cambios en las programaciones de python con el fin de que el HOME pudiese tomar dicho valor para que al ejecutar el hardhome este pudiese ser cambiado mientras se daba la ejecución de un script pero que al finalizar se volvería a setear al valor inicial.

Ya que el problema más importante a resolver fueron las ecuaciones de cinemática inversa se intentó realizar un despeje simbolico utilizando Matlab, este dió como resultado las siguientes ecuaciones para resolver theta2 y theta3 dado que se utilizó alfa=theta4, beta=theta5, gamma=theta4+theta3+theta2 según la literatura[1].

```
A=((X/m.cos(t1))-140*m.cos(t4)-25)/235
B=(Z-410-140*m.sin(t4))/235
C=A**2 + B**2

t21= 2*atan((2*A + (A**2*(-C*(C - 4))**(1/2))/C +
(B**2*(-C*(C - 4))**(1/2))/C)/(2*B + C))

t22= -2*atan(((A**2*(-C*(C - 4))**(1/2))/C - 2*A +
(B**2*(-C*(C - 4))**(1/2))/C)/(2*B + C))

t31 = -2*atan((-C*(C - 4))**(1/2)/C)

t32 = 2*atan((-C*(C - 4))**(1/2)/C)
```

A modo iteración para darle solución se planteó que se podría utilizar otros metodos para resolver la cinemática inversa, por ejemplo la implementación de un algoritmo iterativo para acercarse al valor de posición deseado mediante los valores de ángulos pero el mismo requiere que el sistema que lo realice posea suficiente procesamiento dado que al implementarlo en la computadora personal con un procesador de 5 núcleos, la misma se veía afectada en terminos de ren-

dimiento de tal manera que no se recomienda porque la tabla obtenida tiene una resolución de 2° con 28 000 000 000 de puntos por lo que no es recomendado mediante dicha tabla de posibles puntos.

Finalmente cabe destacar que se logró la redacción de un paper con formato IEEE para su publicación, en el mismo se comparte el alcance del proyecto con el fin de presentar la metodología seguida para la actualización de un robot scorbot a nivel educativo y qué pautas se tomaron en cuenta para cumplir con los requisitos planteados, asimismo presenta las dificultades más importantes que se presentaron al realizar un proyecto como este de bajo costo a nivel profesional e intentar cumplir con el objetivo de actualización del mismo.

Capítulo 7

Conclusiones y recomendaciones

7.1. Conclusiones

- El controlador, los motores y sensores permiten cumplir satisfactoriamente con la actualización planteada.
- Se reestableció mecánicamente el brazo robótico para minimizar la variación angular.
- Se logró un control de las acciones y programación de la interfaz mediante la implementación de VNC entre el usuario remoto y el sistema de control general del brazo robótico Scorbot ER-4U.
- Se logró la redacción de un paper de estilo IEEE y se postuló para la convención de estudiantes CONESCAPAN 2019 a realizarse en San José, Costa Rica con el fin de compartir la metodología llevada a cabo para la actualización del control del brazo robótico Scorbot ER-4U.
- La implementación de la estructura maestro-esclavo permite una comunicación efectiva entre el control y la interfaz.
- Se logró obtener una interfaz que cumple con los requisitos de robustez esperados para darle control al brazo robótico del Instituto Tecnológico de Costa Rica Campus Tecnológico Local San Carlos.

7.2. Recomendaciones

- Buscar y/o implementar otro controlador capaz de aumentar el procesamiento de datos actual para que el mismo no se vea afectado al realizar el control, ya sea mediante la implementación de PLC o de algún microcontrolador especifico.
- Mejorar el sistema visualmente, realizando una armazón de algún material más resistente dado que el actual de MDF ya se encuentra deterioriorado.
- Mejorar el sistema de potencia mediante el diseño de circuitos específicos para generar los voltajes para velocidad máxima de los motores(30V) y dar potencia al sistema de sensado así como la implementación de drives que permitan la corriente nominal de los mismos.
- Realizar un sistema de visión mediante cámaras para implementar un sistema neuronal o similar con el fin de dar realimentación más acertada dado que con los sensores actuales no se puede resolver algunos problemas de posicionamiento mientras se encuentra apagado.
- Realizar el cambio del motor A para aumentar la presición que se alcanza y mejorar el control general.
- En el caso de dar continuidad a las matrices cinemáticas, encontrar un método de iteración utilizando las matrices de cinemática directa de tal manera que se pueda conocer el comportamiento del robot mediante ecuaciones y dar la solución con las primeras tres articulaciones.
- Utilizar un programa más robusto para el cálculo de las matrices cinemáticas dado que este puede restringir la capacidad de reacción y control del robot o un sistema específico como ROS.

Bibliografía

[1] F. Reyes, *Robótica - control de robots manipuladores*. ISBN:978:607:707:190-7, México: Alfaomega Grupo Editor, first ed., mar 2011.

[2] A. Ollero B, *Robótica: manipuladores y robots móviles*. Barcelona, España: MARCOMBO S.A., first ed., 2001.

[3] E. Velasco, *Educatrónica: innovación en el aprendizaje de las ciencias y la tecnología*. España: Universidad Nacional Autónoma de México, 2007.

[4] A. Somolinos S, *Modelado dinámico y control de un robot flexible de tres grados de libertad*. 681.51 (043,2), España: Universidad de Castilla-La Mancha, first ed., 2001.

[5] C. S. A, *Neumatica e Hidráulica*. Barcelona, España: MARCOMBO S.A., second ed., 2011.

[6] J. A. J. . F. C. Mandado E., Acevedo, *Autómatas programables y sistemas de automatización*. España: MARCOMBO S.A., second ed., sep 2009.

[7] Amiyoled, "Final de carrera con rueda kw12-3 - 5a 250vac - 1 conmutado - arduino, electrónica.," jan 2019.

[8] M. J. S.A., "Itr8102 interruptor Óptico," aug 2018.

[9] Broadcom, "Two channel optical incremental encoder modules," oct 2018.

[10] J. Craig, *Robótica*. México: Pearson Education, third ed., 2006.

[11] Intelitek, "Scorbot er-4u educational robot," sep 2018.

[12] R. Kumar and P. Chand in *Inverse kinematics solution for trajectory tracking using artificial neural networks for SCORBOT ER-4u*, 02 2015.

[13] B. . B. Moslehpour, Odom, "Scorbot er-iii robot," *the Technology Interface Journal*, vol. 10, no. 3, 2010.

[14] V. Verma, A. &. Deshpand, "End-effector position analysis of scorbot-er vplus robot," *International Journalof Smart Home*, vol. 5, pp. 1–3, jan 2011.

[15] M. Mondada, F. & Ben-Ari, "Elements of robotics," oct 2017.

[16] A. González, *ANÁLISIS CINEMÁTICO Y DINÁMICO DEL ROBOT SCORBOT-ER VPLUS PARA LA NUEVA CONFIGURACIÓN EN UNA BASE DESLIZANTE*. Ingeniero mecánico, UNIVERSIDAD TECNOLÓGICA DE PEREIRA, Pereira, Colombia, aug 2014.

[17] R. V. J. Chacón, M; Sandoval, *Percepción visual - Aplicada a la robótica*. ISBN:978-633-192-1, México: Alfaomega Grupo Editor, first ed., jul 2015.

[18] Aircraft, "Touring machine company: Pitch, roll, and yaw," nov 2007.

[19] A. Higuera and F. García, *CIM, el computador en la automatización de la producción*. CIENCIA Y TÉCNICA, Ediciones de la Universidad de Castilla-La Mancha, 2007.

[20] P. S. U. D. of Mechanical and N. Engineering, "Servo robot experiment," jan 2019.

[21] A. Maryland, "Pegasus robot motion controller project," oct 2018.

[22] E. Robotec, "User's. manual scorbot-er 5plus.," *ATS Advanced Terminal Software version 1.9.*, vol. 3er edition., feb 1998.

[23] Amatrol, "Flexible manufacturing learning system 94-fms- 1 -a," dec 2018.

[24] J. Medina, "Desarrollo de un manipulador didáctico con una cadena cinemática abierta de 6 grados de libertad," maestrÍa sistemas automÁticos de producciÓn pereira, UNIVERSIDAD TECNOLÓGICA DE PEREIRA, Pereira, Colombia, dec 2016.

[25] M. Román, "Kinematic simulator of an industrial robot arm," grado en ingeniería de computadores, ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA, Universidad de Málaga, dec 2014.

[26] E. P. Leite, *Matlab: Modelling, Programming and Simulations*. India: SCIYO, 2010.

[27] U. de Chile, "Modelación cinemática del brazo del manipulador," sep 2018.

[28] J. R. John Hyde, Albert Cuspinera, *Control electroneumático y electrónico*. Barcelona, España: MARCOMBO S.A., 1997.

[29] R. P. Foundation, "Raspberry pi 3 model b," sep 2018.

[30] P. S. Foundation, "Python for beginners," sep 2018.

[31] N. developers, "Numpy," sep 2018.

[32] F. Pedregosa, "2.10. sympy : Matemáticas simbólicas en python," feb 2019.

[33] P. S. Foundation, "math — mathematical functions," jan 2019.

[34] R. Sileika, *Pro Python System Administration*. Expert's voice in open source, Apress, 2014.

[35] A. Kurniawan, *Getting Started with Raspberry Pi 3*. Berlín: PE Press, 2016.

[36] N. Electronics, "Raspberry pi hat - 32 i/o port expander - mcp23017 - i2c - black - kit edition," feb 2019.

[37] M. T. Inc., "Mcp23017 16-bit i2c i/o expander with serial interface," feb 2019.

[38] R. P. Foundation, "Vnc (virtual network computing)," sep 2018.

[39] M. I. of Technology, "Red hat enterprise linux 4: Manual de referencia," sep 2018.

[40] I. Amazon Web Services, "What is aws iot greengrass?," sep 2018.

[41] A. P. E. Sum, "¿qué es gpio?," may 2019.

Apéndice A

Glosario, abreviaturas y simbología

L_1= Longitud del primer eslabón(en mm)

L_2=Longitud del segundo eslabón(en mm)

L_3=Longitud del tercer eslabon(en mm)

L_4=Longitud del Gripper(en mm)

a= Distancia paralela al eje x entre el sistema de referencia 2(base2) y 3 (hombro) (en mm)

d_1=Distancia en el riel en el que se encuentra ubicado el robot(en mm)

θ_1=ángulo formado entre la base y la posición cero de la base(en grados o radianes)

θ_2=ángulo formado entre el hombro y la posición cero del hombro(en grados o radianes)

θ_3=ángulo formado entre el codo y la posición cero del codo(en grados o radianes)

θ_4=ángulo pitch o formado entre la muñeca y la posición cero de la muñeca(en grados o radianes)

θ_5=ángulo de rotación respecto a la posición cero del la muñeca(roll) (en grados o radianes)

X=distancia en el eje definido como x respecto al sistema de coordenadas origen (en mm).

Y=distancia en el eje definido como y respecto al sistema de coordenadas origen (en mm).

Z=distancia en el eje definido como z respecto al sistema de coordenadas origen o altura alcanzada (en mm).

Apéndice B

Manual de usuario

MANUAL DE USUARIO PARA PROGRAMACIÓN Y CONTROL DEL SCORBOT ER 4U

INSTITUTO TECNOLÓGICO DE COSTA RICA

Elaborado: RUTH IVETH CAMPOS ARTAVIA

Diciembre, 2018

Tabla de Contenidos

Movimiento No Interpolado(MOV)

El control robótico se encarga de generar cualquier trayectoria entre el punto actual y el punto de destino(PB), calculando para cada articulación la coordenada final, velocidad y aceleración.

Sintaxis:

MOV PB

Deben indicarse las iniciales MOV seguido de una posición que se encuentre en la base de datos(según como se encuentra en la misma)

Resultado de ejecución: El robot se mueve a la posición indicada(PB) con una interpolación angular de cada eje.

Ejemplo: MOV P1……...Se mueve con una interpolación angular Pactual → P1.

Movimiento de Interpolación Lineal(MVS)

El control genera el movimiento de una línea recta entre el punto actual y el punto final(PB). Es el ideal para movimientos críticos que deben ser lentos y precisos.
Sintaxis:

MVS PB

Deben indicarse las iniciales MVS seguido de una posición que se encuentre en la base de datos(según como se encuentra en la misma)

Resultado de ejecución: El robot se mueve a la posición indicada(PB) con una interpolación lineal.

Ejemplo: MVS P1 ……...Se mueve con una interpolación lineal entre Pactual → P1

Movimiento de Interpolación Circular (MVR/MVR2/MVR3/MVC)

El control genera el movimiento circular entre el punto actual, puntos de tránsito o de no tránsito y el punto final(PB).

Sintaxis:

MVR PA PB

Deben indicarse las iniciales MVR seguido de dos posiciones que se encuentren en la base de datos(según como se encuentra en la misma) PA correspondiente a un punto de tránsito(por donde pasará) y PB el punto de destino.

MVR2 PA PB

Deben indicarse las iniciales MVR2 seguido de dos posiciones que se encuentren en la base de datos(según como se encuentra en la misma) PA correspondiente a un punto de no tránsito(por donde no pasará) y PB el punto de destino.

MVR3 PB PC

Deben indicarse las iniciales MVR3 seguido de dos posiciones que se encuentren en la base de datos(según como se encuentra en la misma) PC correspondiente al centro del círculo y PB el punto de destino.

MVC PA PE PB

Deben indicarse las iniciales MVC seguido de dos posiciones que se encuentren en la base de datos(según como se encuentra en la misma) PA y PE correspondientes a puntos de tránsito(por donde pasará) y PB el punto de destino.

Resultado de ejecución: El robot se mueve a la posición indicada(PB) con una interpolación circular.

Ejemplos:

MVR P1 P2…..……...Se mueve con una interpolación circular entre Pactual → P1 → P2.

MVR2 P1 P2……...Se mueve con una interpolación circular entre Pactual a P2 sin pasar por P1

MVR3 P2 P3……...Se mueve con una interpolación circular entre Pactual a P2, teniendo como centro P3

MVC P1 P4 P2……....Se mueve con una interpolación circular en el siguiente orden Pactual → P1 → P4 → P2

Movimientos del End Effector(HOPEN/HCLOSE)

El control realiza la apertura o cierre de la pinza.

Sintaxis:

HOPEN

Deben indicarse las iniciales HOPEN para la apertura de la pinza

HCLOSE

Deben indicarse las iniciales HCLOSE para el cierre de la pinza

Resultado de ejecución: El robot abre o cierra la pinza de acuerdo con el comando introducido.

Ejemplo:

HOPEN ……...Se abre la pinza

HCLOSE ……...Se cierra la pinza

Movimiento HOME

El control realiza el posicionamiento al punto HOME

Sintaxis:

HOME

Deben indicarse las iniciales HOME.

Resultado de ejecución: El robot se posiciona en la posición definida como HOME

Ejemplo:

HOME ……...Se posiciona en una posición predefinida como HOME.

Parámetros de movimiento general

El control setea algunos parámetros de movimiento como aceleración, desaceleración, velocidad para interpolaciones o velocidad del sistema general

Sintaxis:

ACCEL A D

Deben indicarse las iniciales ACCEL junto con el porcentaje de aceleración(A%) y desaceleración del sistema(D%).

OVDR N

Deben indicarse las iniciales OVDR junto con el porcentaje(N%) de velocidad del sistema.

SPD N

Deben indicarse las iniciales SPD junto con el porcentaje(N%) de velocidad para interpolaciones

Resultado de ejecución: El robot utiliza los parámetros definidos para el movimiento.

Ejemplos:

ACCEL ……..…...Se define aceleración y desaceleración al 100%

ACCEL 10 20 ……...Se define aceleración al 10% y desaceleración al 20%

OVDR 40……...……Se define porcentaje(40%) de velocidad del sistema.

SPD 60……...…Se define porcentaje(60%) de velocidad para ejecución de interpolaciones.

Indice de movimientos

Sobre la aplicación

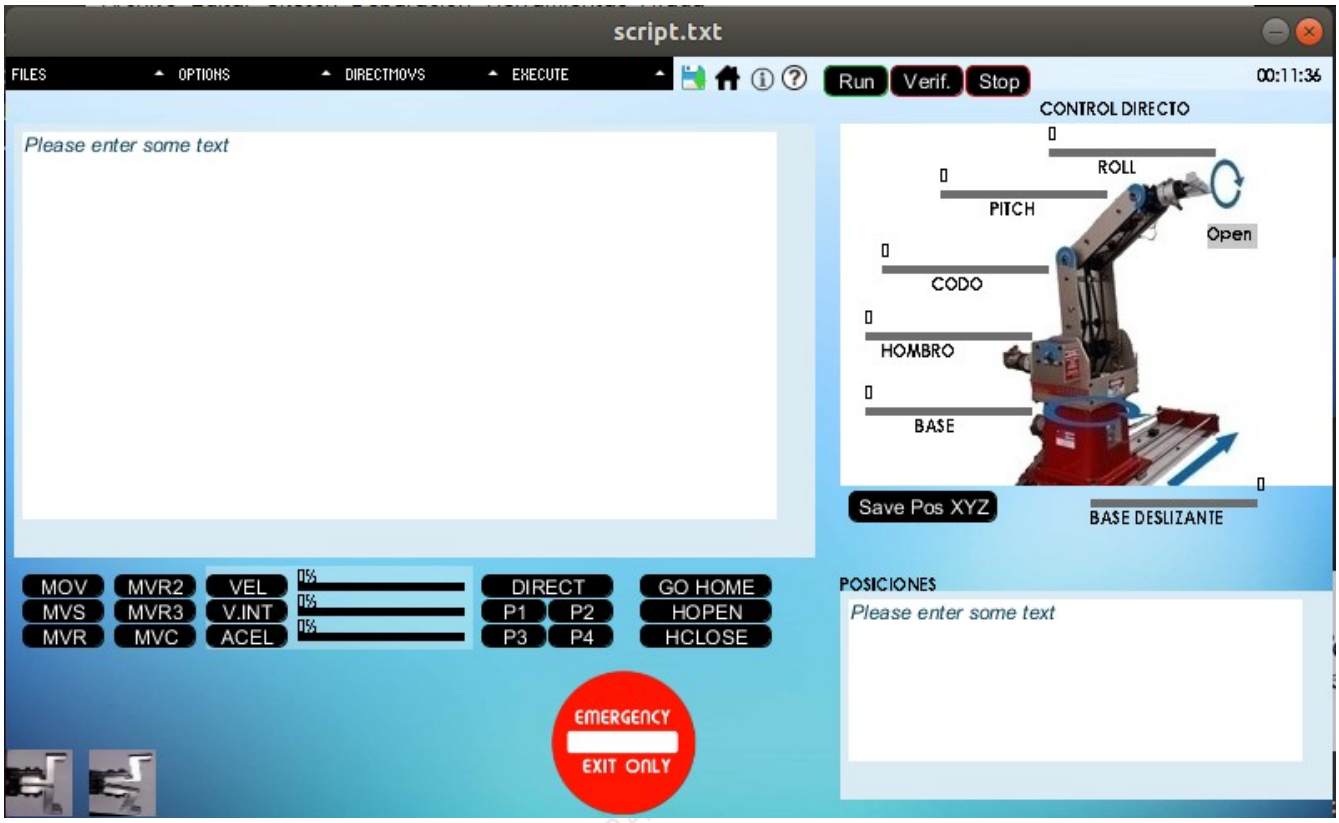


Figura 1. Interfaz de usuario del controlador para el Scorbot ER-4U

1. Panel de acciones miscelaneas:

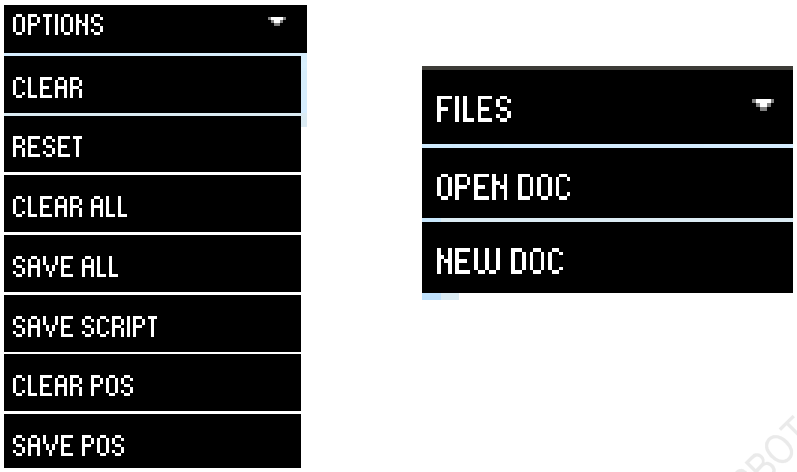


Figura 2. Panel de acciones miscelaneas.

Clear:

Botón para eliminar todos los datos del script de comandos

Reset:

Botón para volver al estado inicial del script abierto.

Clear all:

Botón para eliminar todos los datos tanto del script de comandos como posiciones.

Save all:

Botón para guardar los datos introducidos tanto del script de comandos como posiciones.

Save script:

Botón para guardar los datos introducidos solamente del script de comandos.

Clear Pos:

Botón para eliminar una posición específica del script de posiciones.

Save Pos:

Botón para guardar una posición nueva dada por el punto actual o alguna requerida.

Open Doc:

Botón que abre una ventana emergente para abrir un documento anteriormente creado para obtener los datos del script de comandos.

New Doc:

Botón que crea un documento nuevo donde se puede generar la programación, debe ser guardado al final de la programación.

2. Panel de accesos rápidos.

Figura 3. Panel de accesos rápidos.

El ícono  simboliza el Save all, botón para guardar los datos introducidos tanto del script de comandos como posiciones.

El ícono  es un botón para ir a la posición home de manera directa.

El ícono  contiene información sobre la aplicación como nombre, creación y desarrollo.

El ícono  simboliza el ayuda, al presionar este botón abre un documento .pdf que contiene ayuda general sobre la programación y de la aplicación.

3. Introducción de comandos.

Figura 4. Campo de texto para introducción de comandos.

El cuadro de texto que se muesstra permite la introducción manual de los comandos separados por un ENTER entre ellos.

4. Programación rápida

Figura 5. Barra de botones para programación rápida.

Los botones de la izquierda introducen los comandos mostrados, no obstante, antes de ser introducidos se deben introducir las posiciones de cada movimiento.

Los botones de la derecha están definidos por el botón direct, el que activa o desactiva la ejecución directa de los comandos que se muestran bajo él.

Los botones P1,P2,P3,P4 son cuatro puntos predefinidos, los cuales son puntos para el posicionamiento del end effector o pinza.El posicionamiento es directo en caso que Direct aparezca inactivo(en azul) o lo introduce al script de comandos si el Direct es activado (se encuentra en rojo).

El botón Home realiza la ejecución directa al punto seteado como Home en caso que Direct aparezca inactivo(en azul) o lo introduce al script de comandos si el Direct es activado (se encuentra en rojo).

Los botones HOPEN y HCLOSE correspondientemente abre y cierra la pinza directamente en caso que Direct aparezca inactivo(en azul) o lo introduce al script de comandos si el Direct es activado (se encuentra en rojo)

Los sliders de velocidad, aceleración y velocidad de interpolación permiten el control de la velocidad del sistema, la aceleracion de los motores y la velocidad cuando se realizan acciones de interpolación como MVS,MVR,MVR2,MVR3,MVC.

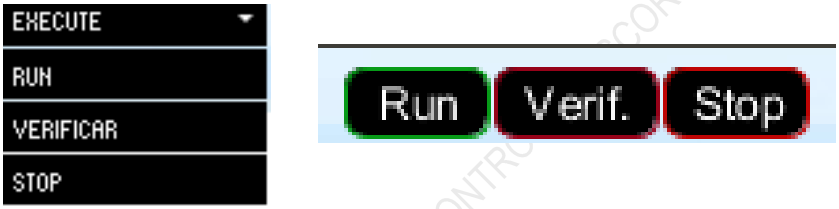


Figura 6.Botones para ejecutar, verificar o parar la ejecución del script de comandos Melfa Basic programados.

5. Control Directo

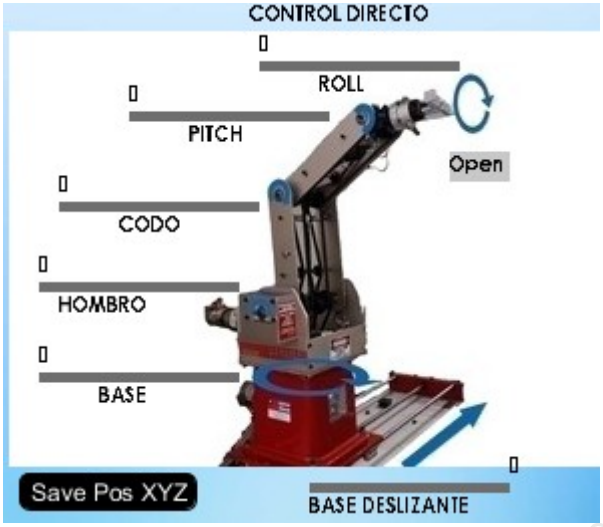


Figura 7. Sección para el control directo del robot

Cada una de las articulaciones puede ser controlada directamente mediante la selección de algún valor mediante los sliders, cada slideer se encuentra limitado por los valores mínimos y máximos que puede tomar cada articulación de manera directa.

El botón ON quiere decir que la pinza se encuentra cerrada, cuando este botón es presionado cambia a OFF lo que quiere decir que la pinza se encuentra abierta y si este es presionado nuevamente se cerrará la pinza nuevamente.

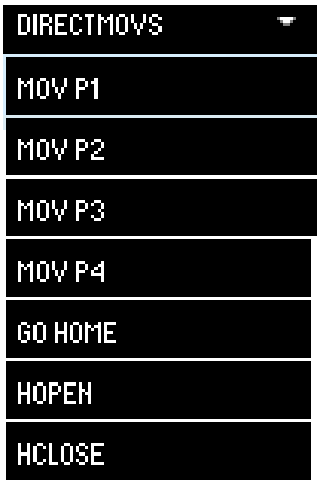


Figura 8. Lista deslizante para el control directo.

La lista deslizante, al igual que los botones designados, realizan las funciones programadas para el control rápido.

Figura 9. Botón de emergencia para para todos los procesos actuales

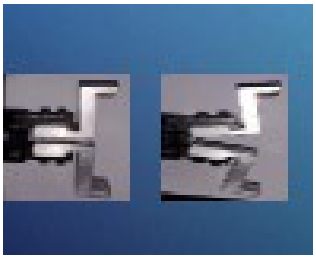


Figura 10.Botones para abrir y cerrar pinza de manera directa.

6. Posiciones

Figura 11. Sección para la visualización e introducción manual de las posiciones

El cuadro de texto que se muesstra con la leyenda Posiciones permite la introducción manual de las posiciones de la forma P unido al número identificador, seguido de las coordenadas (X,Y,Z,alfa,beta,gamma) separadas por un espacio entre ellas.

Apéndice C

Paper generado

The submission has been saved!

Paper 91	
Title:	Methodology for the technological update of the control of the robotic arm Scorbot ER-4U
Paper:	(Jun 03, 04:24 GMT)
Author keywords:	Scorbot ER - 4U interface restoration kinematic matrices control
Abstract:	This document carries out the description of the methodology implemented for updating the current control of the robotic arm Scorbot ER-4U of the Instituto Tecnologico de Costa Rica, San Carlos headquarters in the year 2019. It carried out the implementation of a master-slave structure between Python and processing for the communication of control with the interface created likewise logical tests of the sensors ITR8102 and encoders HEDS9100 and KY-040 and Mechanics for mechanical restoration As the location at zero position and developed a control system using kinematic matrices given by Denavit-Hartenberg, the latter should be improved and it is recommended to implement another controller likewise improve the power system Substituting the power supply by specific circuits for the systems implemented.
Submitted:	Jun 03, 04:24 GMT
Last update:	Jun 03, 04:24 GMT

Authors						
first name	last name	email	country	organization	Web page	corresponding?
Ruth	Campos Artavia	rcampos18@ieee.org	Costa Rica	ITCR		✔

Figura C.1: Figura del paper enviado a la conferencia de estudiantes CONESCAPAN 2019
Fuente: propia

Metodología para la actualización tecnológica del control del brazo robótico Scorbot ER-4U

1st Ruth Campos Artavia
Electronic Engineer Department.
Costa Rica Institute of Technology
Alajuela, Costa Rica
rucampos@ieee.org

***Abstract*—This document carries out the description of the methodology implemented for updating the current control of the robotic arm Scorbot ER-4U of the Instituto Tecnologico de Costa Rica, San Carlos headquarters in the year 2019. It carried out the implementation of a master-slave structure between Python and processing for the communication of control with the interface created likewise logical tests of the sensors ITR8102 and encoders HEDS9100 and KY-040 and Mechanics for mechanical restoration As the location at zero position and developed a control system using kinematic matrices given by Denavit-Hartenberg, the latter should be improved and it is recommended to implement another controller likewise improve the power system Substituting the power supply by specific circuits for the systems implemented.**

***Index Terms*—Scorbot ER - 4U, interface, restoration, kinematic matrices, control**

I. Introducción

El Instituto Tecnológico de Costa Rica (I.T.C.R) cuenta con un brazo robotico Scorbot ER-4U con 6 grados de libertad y una pinza controlada por un sistema neumático que al ser actualizado tecnológicamente y mejorar las características actuales del robot manipulador tanto su control y posicionamiento como su entorno de programación permitiría su utilización para actividades de investigación y docencia por parte de la institución.

La desactualización tecnológica se debe a que inicialmente contó con una unidad de control de la empresa Amatrol, la misma ya no se encuentra disponible en el equipo actual sino que posee un sistema de mando secuencial simple y por ello es necesario un control más robusto para lograr la programación de movimientos y secuencias precisas implementando matrices cinemáticas. Adicionando a lo anterior, la institución no cuenta con fondos suficientes para restablecerlo por parte de la empresa Amatrol o adquirir otro con caracteristicas similares dado que es un equipo de alto valor monetario y su mantenimiento es poco asequible además de que ha sido descontinuado por parte de la empresa fabricante.

Dada la problemática actual, el proyecto pretende aprovechar los recursos e integración de dispositivos y procesamiento disponibles para mejorar el control existente y agregar funciones adicionales como control remoto del robot, restauración física del mismo y un entorno de programación que permita introducir comandos como el Melfa Basic 4 para el control y posicionamiento del brazo en los puntos deseados y este documento realiza una descripción detallada de la metodología para la actualización tecnológica del brazo robótico Scorbot ER-4U así como los resultados de la implementación de dicha tecnología y retos por resolver.

II. Metodología

La metodología que se siguió está seccionada en las siguientes fases: análisis de solucion de materiales, fase de desarrollo de ingeniería, maduración de tecnología, fase de producción de datos e implementación por lo que se presentará un desglose de las mismas para una mejor comprensión del problema y su solución.

A. Análisis de solución de materiales

Primeramente, se realizó un levantamiento de requisistos, seguidamente un análisis de las soluciones potenciales tanto para restaurar el brazo mecánicamente como tecnológicamente y satisfacer los requerimientos operativos fijados inicialmente. En esta fase tambien se identifica y evalúa la asequibilidad de los dispositivos a utilizar para los distintos sistemas del brazo robótico(sistema mecánico, sistema de sensado, sistema de accionamiento, sistema de control, sistema de potencia, interfaz).

Inicialmente se plantearon tres distintas propuestas para el sistema de sensado (ver tabla I), aunque uno de los factores determinantes fue el controlador a utilizar, tener un sistema local permite controlar en menor tiempo las articulaciones del robot por lo que se eligió el control local basado en raspberry. Seguidamente, se analizaron las propuestas para el sistema de accionamiento(ver tabla II) y dado que los motores se encontraban en buen estado, se utilizarían los mismos como parte de la solución entonces las propuestas están basadas en las posibilidades de implementación de los drivers para su control de voltaje de los motores y aunque las dos propuestas son viables, la disponibilidad limita la implementación por lo que se eligió la primera propuesta. Respecto al sistema mecánico se le realizó un mantenimiento que incluyó el desmontaje, la limpieza, el tensado correcto de las cadenas mediante las catarinas y engranajes.

Para el sistema de control, se analizaron posibles controladores para plantear las distintas propuestas según la tabla

TABLE I: Tabla comparativa de propuestas de los distintos sistemas de sensado a implementar en la solución

Características del sistema de sensado	Propuesta 1:Remoto Raspberry	Local Raspberry	Remoto Arduino
Tiempo de procesamiento	ALTO	BAJO	MEDIO
Complejidad	ALTO	MEDIO	FACIL
Programación	C	PYTHON	C++
Velocidad de comunicación	MEDIA	RÁPIDA	LENTA
Dependencia de red	SI	NO	SI

TABLE II: Tabla comparativa de propuestas de los distintos sistemas de accionamiento a implementar en la solución

Características del sistema de accionamiento	Uso 3 puentes H duales	Diseño de un circuito con 6 puentes H
Tiempo de diseño	ALTO	BAJO
Complejidad	BAJA	ALTA
Disponibilidad	SI	NO

III que da un desglose general de las propuestas de solución al sistema general donde la propuesta 1 es la seleccionada ya que los materiales se encuentran disponibles en la institución además que permite la aplicación de cualquier tecnología para la solución y la cantidad de pines a utilizar son más accesibles de manera local por la raspberry que por un arduino o un sistema con PLC's.

En la tabla IV se analizan y comparan tres propuestas para el sistema de potencia, entre ellas una batería, una fuente de poder o un panel solar con una batería, la fuente de poder fue la solución seleccionada ya que presentaba un suministro de varios voltajes además de que se encontraba disponible para su implementación y finalmente permite menor suministro de tiempo aplicado para la solución.

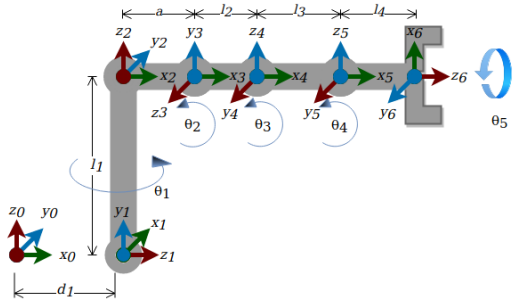


Fig. 1: Representación del modelo cinemático del robot Scorbot ER-4U

B. Fase de desarrollo de ingeniería

En la tabla V se muestran las longitudes del Scorbot ER-4u para la aplicación de las matrices cinemáticas mediante el método de Denavit-Hartenberg(DH) [10]. Este último define la matriz de transformación de rotación respecto al eje $x_i(alpha_i)$, traslación respecto al eje $x_i(a_i)$,rotación respecto al eje $z_i(theta_i)$, traslación respecto al eje $z_i(d_i)$ como se muestra en la tabla VII, la misma toma en cuenta cada uno de los seis grados de libertad(cantidad de un solo eje de rotación de las articulaciones) siendo el conjunto 0-1 los parámetros de traslacion y rotacion de la base deslizante respecto a la base1, 1-2 de la base1 a base2, 2-3 de base2 al hombro, 3-4 del hombro al codo, 4-5 del codo a la muñeca(pitch) y finalmente 5-6 de la muñeca(pitch) a la muñeca(roll) obtenidos a partir de realizar el análisis cinemático del sistema que se muestra en la figura 2, este considerando el análisis obtenido en la tabla VI con la figura 1. Se realizó la implementación los parámetros para obtener el modelo cinemático mediante la implementación de las matrices directas correspondientes tal como se muestra a continuación:

$$A_{i-1}^{i} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & a \\ \cos\alpha\cdot\sin\theta & \cos\alpha\cdot\cos\theta & -\sin\alpha & -d\cdot\sin\alpha \\ \sin\alpha\cdot\sin\theta & \sin\alpha\cdot\cos\theta & \cos\alpha & d\cdot\cos\alpha \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

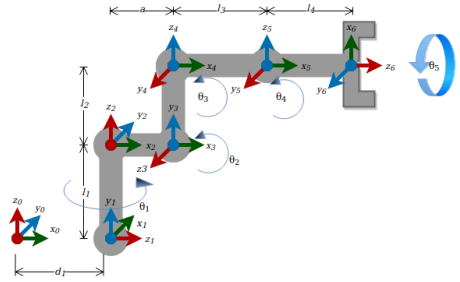


Fig. 2: Modificación del modelo cinemático para establecer el punto HOME del sistema

Se realizó una aproximación de los puntos de movimiento X, Y, Z de la siguiente manera:

$$X = (l_3\cdot\cos(\theta_3) - l_2\cdot\sin(\theta_2) + l_4\cdot\cos(\theta_4) + a) * cos(\theta_1)) \quad (2)$$

$$Y = (l_3\cdot\cos(\theta_3) - l_2\cdot\sin(\theta_2) + l_4\cdot\cos(\theta_4) + a)\cdot\sin(\theta_1) + d1 \quad (3)$$

$$Z = l_3 \cdot sin(\theta_3) + l_2 \cdot \cos(\theta_2) + l_4 \cdot \sin(\theta_4) + l_1 \quad (4)$$

TABLE III: Tabla comparativa de las características de tres distintas propuestas para el controlador de la solución

Características del controlador	Propuesta 1:RASP	Propuesta 2: PC-PLC	Propuesta 3: Arduino
Conexión Remota	SI	NO	SI
Flexibilidad a cambios	ALTA	BAJA	MEDIA
Programación	DIFICIL	FÁCIL	FÁCIL
Navegación	FÁCIL	MEDIA	DIFICIL
Velocidad de procesamiento de datos	ALTA	BAJA	BAJA
Costo	BAJO	ALTO	ALTO
Potencia	MEDIA	BAJA	MEDIA
Sistema Robusto	MEDIA	ALTA	BAJA
Interfaz	SI	SI	SI

TABLE IV: Tabla comparativa de propuestas para el sistema de potencia de la solución

Sistema Potencia	Propuesta 1 Batería	Propuesta 2 Fuente de Poder AT	Propuesta 3 panel solar con batería
Ahorro	SI	NO	SI
Suministro de distintos voltajes	NO	SI	NO
Disponibilidad	NO	SI	NO

TABLE V: Longitudes del robot SCORBOR ER 4U a rehabilitar

Articulación	Base Deslizante(d_1)	Base(L_1)	Hombro(L_2)	Codo(L_3)	Pitch/roll(L_4)	a
Medida[mm]	1000	410	230	230	150	25

TABLE VI: Parámetros del robot SCORBOR ER 4U para el análisis cinemático. 1

	alfa	a	theta	d
0-1	$\pi/2$	0	0	$-d_1$
1-2	$-\pi/2$	0	$\theta_1-\pi/2$	L_1
2-3	$\pi/2$	a	θ_2	0
3-4	0	L_2	θ_3	0
4-5	0	L_3	θ_4	0
5-6	$\pi/2$	0	$\pi/2+\theta_5$	L_4

TABLE VII: Parámetros del robot SCORBOR ER 4U modificada al definir la posición HOME

	alfa	a	theta	d
0-1	$\pi/2$	0	0	$-d_1$
1-2	$-\pi/2$	0	θ_1	L_1
2-3	$\pi/2$	a	$\theta_2+\pi/2$	0
3-4	0	L_2	$\theta_3-\theta_2-\pi/2$	0
4-5	0	L_3	$\theta_4-\theta_3$	0
5-6	$\pi/2$	0	$\pi/2+\theta_5$	L_4

C. Maduración de tecnología

Tal como se indica en la tabla VIII se generaron tres propuestas que fueron implementadas, la interfaz mediante TKinter no proporcionaba la robustez y gráfica que se quería implementar por lo que se intentó realizar una interfaz en Matlab pero se encontró que la raspberry utilizada no poseía una compatibilidad con la misma ya que se realizó una investigación sobre el tipo de programas que permite ejecutar el controlador seleccionado y finalmente se planteó el uso de matlab para simulaciones con un computador convencional capaz de ejecutar dicho programa, con el fin de comparar los datos obtenidos con el resultado final y así realizar correcciones tanto en los algoritmos de código necesarias como para encontrar posibles mejoras del control. Finalmente se seleccionó el programa Processing, de tal manera que la interfaz final es la que se muestra en la figura 3, y se utilizó una configuración maestro esclavo, siendo la interfaz un cliente del control(servidor) para la asequibilidad de los datos y peticiones del usuario. Ya que uno de los requerimientos es la posibilidad de control remoto del brazo robótico, el sistema controlado necesitó la implementación de una plataforma de uso compartido de escritorios y el programa seleccionado e implementado para el control del brazo remotamente fue VNC, el mismo consta de una interfaz que permite el intercambio de imagenes entre el escritorio controlado y el servidor o controlador remoto. Adicional a lo anterior, se realizó el uso de Processing en modo Java para realizar la interfaz gráfica ya que permite obtener una interfaz de alta calidad y una velocidad considerable de captación de datos entre el usuario final y el control del sistema.

Se implementó el programa python tanto para la programación del control general y se implementó una comunicación maestro-esclavo entre los distintos módulos creados como para la comunicación entre la interfaz y el control general y el uso del programa VNC para la conexión remota de usuarios.

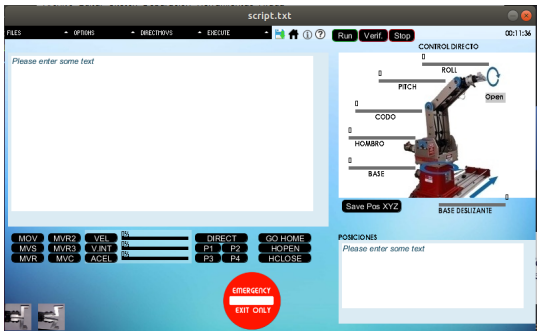


Fig. 3: Interfaz elaborada con Processing 3.4 para control directo o indirecto del brazo robótico

TABLE VIII: Tabla comparativa de propuestas para el sistema de interfaz de la solución

Interfaz	Propuesta 1 Python Tkinter	Propuesta 2 Processing	Propuesta 3 Matlab
Complejidad	BAJA	MEDIA	ALTA
Velocidad	LENTO	RÁPIDO	RÁPIDO
Simulación y Gráficos	SI	SI	SI
Compatibilidad controlador	SI	SI	NO

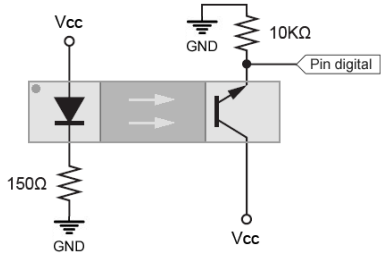
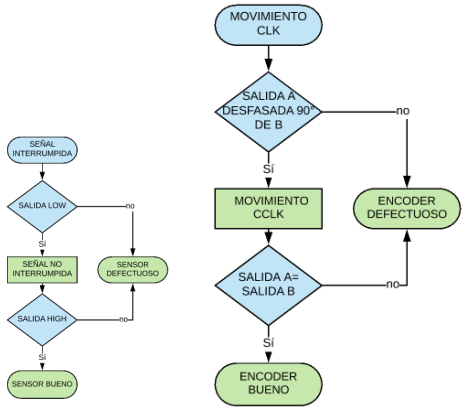


Fig. 5: Circuito para sensado mediante el interruptor óptico ITR8102

D. Fase de producción de datos e implementación

Algunas de las pruebas realizadas para la restauración del brazo robótico son las que se describen en las figura 4a y 4, en la izquierda se muestra la prueba lógica realizada a cada uno de los sensores ópticos ITR8102 implementados según el circuito de la figura 5.



(a) Sensores (b) Encoders

Fig. 4: Prueba lógica para la revisión del buen funcionamiento de los encoders y sensores ópticos

Se suministró potencia mediante circuitos no específicos, mediante una fuente de poder, la misma dado que solo es utilizada para dar abastecimiento a los motores y electrificar el sistema de sensado no se encontró útil y se recomienda la sustitución de la misma por circuitos específicos de 12V, 5V y 3.3V para los motores serie GM94, los drivers para

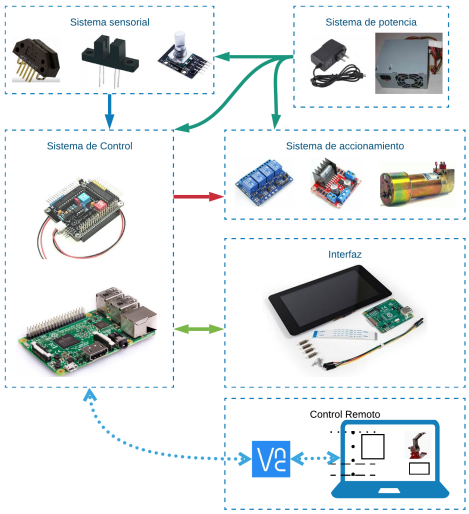


Fig. 6: Diagrama de solución implementada para la mejora tecnológica del brazo robótico Scorbot ER-4U

los motores(puentes H duales L298) y el sistema de sensado(ITR8102, KY-040 Y HEDS9100) respectivamente como se muestra en el diagrama de la solución implementada de la figura 6. Al implementar el movimiento directo que se programó, se encontraron varios problemas, entre ellos que el conteo de los encoders no es el óptimo, esto se puede deber al estado de los mismos porque son los integrados inicialmente por la empresa fabricante del brazo robótico por lo que es necesario implementar un hard home inicial y que el mismo sea ubicado en un lugar estratégico.

Adicionalmente se encuentra un problema respecto a que el control del mismo es bastante robusto y necesita un procesamiento mayor al adquirido por la raspberry pi 3 porque se implementaron paralelismo(hilos o threads) para el movimiento de cada uno de los motores, para la lectura del documento con los ángulos finales y actuales, entre la conexión maestro-esclavo implementada utilizando la interfaz(desarrollada en Processing) y el servidor que es el controlador principal del sistema, la actualización de la matriz rotacional de cinemática directa y la obtención de los parámetros de la cinemática inversa, los distintos tipos de movimientos de los comandos.

Al implementar el movimiento por comandos, el programa es capaz de leer los comandos correctamente, insertar posi-

ciones deseadas mediante la interfaz gráfica y posteriormente verificar antes de realizar una ejecución, esto con el fin de que las ejecuciones sean las correctas, partiendo del HOME. Para los movimientos desde un archivo de comandos se implementó la conexión entre archivos de texto con el fin de tener acceso desde el servidor y que puedan ser procesados por el servidor para el control de manera correcta, se logró la lectura de los mismos mas no la implementación de la cinemática inversa ni movimientos, por lo que se recomienda que al darle seguimiento al mismo se obtenga el porcentaje de repetibilidad (de precisión) de movimientos con comandos básicos del lenguaje Melfa Basic 4 del brazo robótico SCORBOT ER-4U.

Además es necesario realizar una implementación de Jacobianos con el fin de dar solución a la velocidad del movimiento de los motores y con ello mejorar la respuesta. Se creó un programa de control que permita ser utilizado en cualquier tipo de sistema operativo ya que python posee la factibilidad de conectarse como servidor o cliente, esto permite que la solución generada sea flexible y facil de implementar en otro tipo de arquitecturas que posean el procesamiento necesario, igualmente al ser Processing un programa que permite crear ejecutables, no es necesario la instalación del programa Processing para la integración de la interfaz gráfica como parte de la solución de control [16].

Finalmente se recomienda que se analicen otros controles que no fueron implementados como el control mediante aprendizaje computarizado, redes neuronales, vision computarizada e implementar otro tipo de controlador con el fin de dar mayor procesamiento, un control más robusto y fluido dado que la raspberry no logra procesar adecuadamente el control ni dar un control instantáneo sobre las variables controladas, lo que tambien se ve influido por el la conexión a internet utilizada de la misma para el control a traves de la plataforma VNC, el exceso de variables y datos a controlar al emplear el expansor de 32 pines de Raspberry Pi HAT, MCP23017 mediante I2C lo que puede exceder la capacidad de procesamiento y provocar a mediano plazo un deterioro del controlador.

***Abstract*—En este documento se realiza la descripción de la metodología implementada para la actualización del control actual del brazo robotico Scorbot ER - 4U del Instituto Tecnológico de Costa Rica, Sede San Carlos en el año 2019. Se llevó a cabo la implementación de una estructura maestro-esclavo entre Python y Processing para la comunicación del control con la interfaz creada asímismo pruebas lógicas de los sensores ITR8102 y encoders HEDS9100 y KY-040 y mecánicas para la restauración mecánica como la ubicación en la posición cero y se desarrolló un sistema de control utilizando matrices cinemáticas dadas por Denavit-Hartenberg, este último debe ser mejorado y se recomienda la implementación de otro controlador asímismo mejorar el sistema de potencia sustituyendo la fuente de poder por circuitos específicos para los sistemas implementados.**

***Index Terms*—Scorbot ER - 4U, interfaz, restauración, matrices cinemáticas, control**

REFERENCES

[1] Amatrol. Flexible manufacturing learning system 94-fms- 1 -a, dec 2018.

[2] S. Amatya and S. Petchartee. Real time kinect based robotic arm manipulation with five degree of freedom. In *2015 Asian Conference on Defence Technology (ACDT)*, pages 1–6, April 2015.

[3] A. N. Barakat, K. A. Gouda, and K. A. Bozed. Kinematics analysis and simulation of a robotic arm using matlab. In *2016 4th International Conference on Control Engineering Information Technology (CEIT)*, pages 1–5, Dec 2016.

[4] Broadcom. Two channel optical incremental encoder modules, oct 2018.

[5] R.; Vega J. Chacón, M; Sandoval. *Percepción visual - Aplicada a la robótica*. ISBN:978-633-192-1. Alfaomega Grupo Editor, México, first edition, jul 2015.

[6] Universidad de Chile. Modelación cinemática del brazo del manipulador, sep 2018.

[7] O. Hock and J. Sedo. Inverse kinematics using transposition method for robotic arm. In *2018 ELEKTRO*, pages 1–5, May 2018.

[8] R. R. Kumar and P. Chand. Inverse kinematics solution for trajectory tracking using artificial neural networks for scorbot er-4u. In *2015 6th International Conference on Automation, Robotics and Applications (ICARA)*, pages 364–369, Feb 2015.

[9] J. Medina. Desarrollo de un manipulador didáctico con una cadena cinemática abierta de 6 grados de libertad. Maestría sistemas automáticos de producción, Universidad Tecnológica de Pereira, Pereira, Colombia, dec 2016.

[10] Mohammed Abu Qassem, I. Abuhadrous, and H. Elaydi. Modeling and simulation of 5 dof educational robot arm. In *2010 2nd International Conference on Advanced Computer Control*, volume 5, pages 569–574, March 2010.

[11] M. Mondada, F. & Ben-Ari. Elements of robotics, oct 2017.

[12] Pennsylvania State University Department of Mechanical and Nuclear Engineering. Servo robot experiment, jan 2019.

[13] M. G. Papoutsidakis, D. D. Piromalis, and G. E. Chamilothoris. Modern control interface for scorbot er-iii robot. In *2014 International Symposium on Fundamentals of Electrical Engineering (ISFEE)*, pages 1–5, Nov 2014.

[14] F. Reyes. *Robótica - control de robots manipuladores*. Alfaomega Grupo Editor, 2011.

[15] M. Román. Kinematic simulator of an industrial robot arm. Grado en ingeniería de computadores, Escuela Técnica Superior de Ingeniería Informática, Universidad de Málaga, dec 2014.

[16] R. Szabo and A. Gontean. Scorbot-er iii robotic arm control with fpga using image processing with the possibility to use as them as sun trackers. In *2017 40th International Conference on Telecommunications and Signal Processing (TSP)*, pages 563–566, July 2017.

Apéndice D

Información sobre la institución

En este apartado se realiza una descripción y antecedentes del proyecto desarrollado.

D.1. Descripción de la institución

Nombre: Instituto Tecnológico de Costa Rica.

Campus Tecnológico Local San Carlos.

Ubicación: Santa Clara, Florencia, San Carlos.

D.2. Descripción del departamento

El proyecto se encuentra desarrollado por parte de la Escuela de Ingeniería Electrónica en cooperación con la Escuela de Producción Industrial, sin embargo el usuario final serán los docentes e investigadores de la institución educativa asímismo los estudiantes de Ingeniería en Producción Industrial.

D.3. Antecedentes Prácticos

Los antecedentes prácticos están compuestos por la implementación de un programa de manera secuencial que fue desarrollado anteriormente por los estudiantes del Instituto Tecnológico de Costa Rica, de la carrera de Ingeniería Electrónica como parte del curso de estructura de microprocesadores.