Prototipo para el control a distancia de un brazo robot construido en un entorno virtual con software de distribución libre

Daniel Andrés Ibarguen Mosquera

Asesor

Andrés Alejandro Diaz Toro

Universidad Nacional Abierta y a Distancia UNAD

Escuela de Ciencias Básicas, Tecnología e Ingeniería ECBTI

Ingeniería Electrónica

2024

Agradecimientos

En primer lugar, quiero agradecerle a Dios creador del universo y a mis padres por haberme dado la vida y los recursos para salir adelante en estos años de vida, especialmente a Dios por ayudarme a derribar cada obstáculo que se presentó en mi proceso formativo y por haber puestos las personas correctas en mi camino para llevar a cabo mis sueños.

A mi padre Franklin Ibarguen, mi madre Yenni Amparo Mosquera, mi hermana Ashley Ibarguen Mosquera y mi sobrino Isaac Ibarguen los cuales son el motivo de mi esfuerzo y que esto sea para brindarles una vida mejor.

A Yuri Alexandra palacios Montenegro, mi actual pareja por cada una de sus asesorías y técnicas con respecto a su opinión sobre este trabajo.

En tercer lugar, a la Universidad Nacional Abierta y A Distancia UNAD, por haber puesto un excelente plantel de profesionales el cual me llevo a fortalecer mis conocimientos y a formarme como profesional. En especial quiero agradecer Al tutor Andrés Alejandro Diaz, por cada una de las asesorías y por ayudarme a entender de una mejor manera la forma de modelar correctamente un Brazo Robot Industrial. A la psicóloga Madeleyne Quintero Rodríguez, al Ingeniero Mario Ramos por su excelente gestión y a todo el personal de la sede Udyr Cali.

En cuarto lugar, agradezco a los desarrolladores de los softwares Blender, Kicad. Flatcam, Fusion 360 y el lenguaje de programación Python. Estos fueron los softwares utilizados para el desarrollo del proyecto, de antemano un abrazo por ayudarme a alcanzar un logro que quizás muchos en el pasado no tuvieron estas herramientas gratuitas para lograrlos.

Resumen

Los robots manipuladores tienen un rol muy importante a nivel mundial en la revolución tecnológica actual. Colombia es un país clave considerando que países asiáticos y europeos están trasladando sus fábricas a regiones de Latinoamérica, de manera que es necesario incentivar el aprendizaje y control de los brazos robots. En este proyecto de grado se da a conocer un software capaz de controlar el movimiento de un brazo robot empleando herramientas de código abierto para el aprendizaje de la cinemática directa, herramientas gráficas como OpenGL y lenguaje de programación Python. Como características principales del sistema se destaca el control inalámbrico de la posición y orientación de la cámara virtual, de las rotaciones de los eslabones del robot, y que el sistema es capaz de calcular la ubicación del efector final del robot utilizando el algoritmo de Denavit Hartenberg.

Palabras clave: Brazo Robot, Python, OpenGL, Control inalámbrico, Denavit Hartenberg

Abstract

Manipulator robots have a very important role worldwide in the current technological revolution. Colombia is a key country considering that Asian and European countries are moving their factories to Latin American regions, so it is necessary to encourage learning and control of robot arms. This degree project presents software capable of controlling the movement of a robot arm using open source tools for learning direct kinematics, graphic tools such as OpenGL and the Python programming language. The main features of the system include wireless control of the position and orientation of the virtual camera, the rotations of the robot links, and the system is capable of calculating the location of the final effect of the robot using the Denavit Hartenberg algorithm.

Keywords: Robot Arm, Python, OpenGL, Wireless Control, Denavit Hartenberg

Tabla de Contenido

Introducción 18

Problema de Investigación 19

Objetivos 23

Objetivo General 23

Objetivo Específicos 23

Justificación 24

Marco Conceptual y Teórico 27

Robot Manipulador 29

Brazos Mecánicos 29

Robot Industrial RV-2AJ 29

Trabajos Relacionados 31

Controlador Electrónico de Software Libre para la Operación del Brazo Robótico Scorboter_4u 31

Diseño y Desarrollo de un Simulador de Código Abierto para un Robot Submarino de Propósito General 31

Navegación de Robots Manipuladores en el Entorno de ROS 32

Webots 32

CoppeliaSim 32

MRPT 32

OpenGL 33

Matrices de Escala 35

Matriz de Rotación 36

Matriz de Traslación 37

ESP8266 40

PIC16F877A 40

Interfaz de Comunicación Serial UART 41

Metodología 45

Enfoque de Investigación 45

Alcance de Investigación 45

Diseño de Investigación 46

Instrumentos de Investigación 46

Cronograma de Actividades 47

Construcción del Brazo Robot en el Software de Distribución Libre Blender 49

Sistema de partes 52

Base 52

Antebrazo Brazo Robot NJT-23 54

Brazo del Robot NJT-23 55

Nota. a. rotación dependiente b. rotación independiente *Fuente.* Elaboración propia Muñeca del Brazo Robot 57

Algoritmo de Denavit-Hartenberg 60

Formar las matrices Homogéneas. 65

Articulación Q1 66

Articulación Q2 66

Articulación Q3 68

Articulación Q4 69

Programa de Escritorio y Renderizado de Gráficos 72

Widgets en Qt Designer 74

Diseño de Interfaz 76

OpenGL 77

Diseño del Circuito Transmisor 85

PIC16F877A: 85

Puerto ADC PIC16F877A 86

Temporizador de 8 Bits Timer0 del PIC16F877A 87

Modo Contador 88

Modo Temporizador 88

Entradas y Salidas Digitales del PIC16F877A 89

Puerto Serie Síncrono/Asíncrono USART 89

Joysticks 90

Botones para la Elección de la Articulación del Brazo Robot, Reseteo de Cámara y

Articulaciones..... 95

Potenciómetro Digital 98

Funcionamiento de un Potenciómetro Digital Rotativo Enconder..... 98

Chip para la transmisión de datos mediante el uso del ESP8266..... 101

Circuito Transmisor..... 104

Diseño PCB Del Circuito Transmisor..... 105

Modelo 3D Del Circuito PCB 106

Especificaciones técnicas transmisor: 108

Diseño del Módulo Receptor 110

Circuito Receptor 113

Esquema Circuito Receptor..... 114

Modelo 3D del circuito Receptor..... 115

Especificaciones..... 116

Pruebas Finales e Información sobre el Software..... 117

Pruebas software del PC..... 117

Pyinstaller..... 117

Consumo de Recursos del Software de Computadora..... 119

Pruebas de Movimiento de la Cámara Mediante el Teclado. 119

Pruebas de Rotación de la Cámara: 122

Prueba de Ventana Informativa Modelo del Brazo Robot NJT23..... 123

Prueba de Funcionamiento Widgets Informativos en Tiempo Real del Programa 126

Posición del Efector Final 126

Valores de Angulo de Rotación de cada Articulación 127

Widget Estado del Programa. 127

Conexión y Prueba del Circuito o Modulo Receptor 128

Pasos para Verificación de Conexión del Receptor: 128

Conexión y Prueba del Circuito Transmisor 131

Prueba Joystick de Traslación de Cámara 134

Prueba del Joystick de Rotación de Cámara 137

Prueba de Botones y Selección de Articulación 139

Prueba del Potenciómetro Digital Encoder 141

Bibliografía 147

Lista de Tablas

Tabla 1 *Cronograma de Actividades del Proyecto* 47

Tabla 2 *Tabla de Longitudes y Alcance en Grados de las Articulaciones*..... 51

Tabla 3 *Identificadores de Eslabones y Articulaciones* 51

Tabla 4 *Tabla de Parámetros Algoritmo Denavit Hartenberg* 65

Tabla 5 *Parámetros Articulación Q1* 66

Tabla 6 *Parámetros Articulación Q2* 67

Tabla 7 *Parámetros Q3*..... 68

Tabla 8 *Parámetros Q4*..... 69

Tabla 9 *Tabla de Descripción Widgets Qtdesigner* 74

Tabla 10 *Especificaciones Técnicas Hardware Transmisor de Eventos* 109

Tabla 11 *Especificaciones Hardware Receptor de Eventos* 116

Lista de Figuras

Figura 1 *Software Robot Estudio Empresa ABB* 19

Figura 2 *Análisis Cuantitativo Unidades de Robots Industriales Vendidos Multipropósito desde el Año 2016 al 2021* 21

Figura 3 *Árbol de Causa y Efecto* 22

Figura 4 *Modelado de un Circuito Electrónico Utilizando el Software Libre KICAD* 25

Figura 5 *Software Libre Blender en su Versión 2.90.1* 26

Figura 6 *Robot Elmer Diseñado por William Grey Walter* 28

Figura 7 *Robot Asimo de la Empresa Honda* 28

Figura 8 *Brazo Robot RV-2AJ Ensamblado por la Empresa Festo* 30

Figura 9 *Logo Biblioteca OpenGL* 33

Figura 10 *Renderización de Gráficos 3D con OpenGL* 34

Figura 11 *Renderización de Gráficos 3D con OpenGL Programa F3d* 35

Figura 12 *Escalamiento de un Vector* 36

Figura 13 *Rotación de un Vector* 37

Figura 14 *Traslación de Un Vector* 38

Figura 15 *Python y Qtdesigner* 39

Figura 16 *Diagrama de Pines PIC16F877A* 40

Figura 17 *Conexión PC - Microcontrolador* 41

Figura 18 *Conexión Microcontrolador - Microcontrolador* 42

Figura 19 *Grafica de Datos Protocolo Serial UART* 42

Figura 20 *Diagrama de Bloques del Sistema Utilizando Sistema de Comunicación ESPNOW* 43

Figura 21 *Diagrama Grafico* 44

Figura 22 *Diseño de Brazo Robot con 4 Grados de Libertad Utilizando el Software Blender*.. 50

Figura 23 *Despiece del Robot NJT-23*..... 50

Figura 24 *Modelo 3D Base Robot NJT23*..... 52

Figura 25 *Sistema de Referencia Base del Brazo Robot*..... 53

Figura 26 *Eje de Rotación de la Base*..... 53

Figura 27 *Antebrazo del Robot NJT23* 54

Figura 28 *Eje de Rotación del Antebrazo del Robot NJT 23*..... 54

Figura 29 *Rotación Base y Antebrazo*..... 55

Figura 30 *Brazo Robot NJT-23*..... 56

Figura 31 *Ubicación el Eje Central Articulación 3*..... 56

Figura 32 *Rotación Dependiente e Independiente*..... 57

Figura 33 *Muñeca o Brida del Brazo Robot NJT-23*..... 58

Figura 34 *Jerarquía Articulaciones*..... 58

Figura 35 *Posicionamiento Eje de Rotación Aticulación 4* 59

Figura 36 *Rotación Dependiente e Independiente Articulación Muñeca* 60

Figura 37 *Identificación de Eslabones y Articulaciones*..... 61

Figura 38 *Paso 3 y 4* 62

Figura 39 *Paso 5*..... 63

Figura 40 *Paso 6, 7 y 8* 64

Figura 41 *Logo Qtdesigner*..... 72

Figura 42 *Pantalla de Inicio en Qtdesigner*..... 73

Figura 43 *Diseño de Interfaz para la Ejecución de la Simulación de Movimiento del Brazo Robot*..... 76

Figura 44 *Triangulo con 3 Vértices* 77

Figura 45 *Figura 3D Compleja Creada a partir de Varios Puntos de Vértices* 78

Figura 46 *Canalización de Gráficos OpenGL* 79

Figura 47 *Representación en Bytes de 3 Vértices en Memoria* 80

Figura 48 *Ejemplo de Contenido Archivo Obj* 81

Figura 49 *Algoritmo para Obtener Datos del Modelo* 82

Figura 50 *Diagrama de Bloques Metodología Objetivo Numero 2* 83

Figura 51 *Resultado Metodología 2* 84

Figura 52 *PIC16F877A DIP 40* 86

Figura 53 *Diagrama de Bloques Convertidor ADC* 87

Figura 54 *Configuración de Resistencias Pull Up y Pull Down* 89

Figura 55 *Transmisión y Recepción de Datos Serie Puerto USART* 90

Figura 56 *Modelo de Modulo Joystick* 91

Figura 57 *Sensor de Posición Resistivo* 91

Figura 58 *Valores de los Ejes X y Y del Joystick en los Canales ADC del Microcontrolador* ... 92

Figura 59 *Vista del Brazo Robot desde Diferentes Puntos de la Cámara* 93

Figura 60 *Algoritmo Lectura Canal ADC* 94

Figura 61 *Pulsadores de 2 Pines* 95

Figura 62 *Algoritmo Lectura Botones Articulación* 96

Figura 63 *Algoritmo para Restablecer Cámara y Articulaciones* 97

Figura 64 *Potenciómetro Rotativo Digital* 98

Figura 65 *Funcionamiento de un Potenciómetro Encoder* 99

Figura 66 *Algoritmo de Lectura Potenciómetro Encoder* 100

Figura 67 *Esp8266 CHIP*..... 101

Figura 68 *ESP8266 Wemos Mini*..... 102

Figura 69 *Diagrama de Bloques Metodología Objetivo Específico Numero 3*..... 103

Figura 70 *Esquemático Circuito Transmisor*..... 104

Figura 71 *Pcb Circuito Transmisor*..... 105

Figura 72 *Modelo 3D Control Remoto*..... 106

Figura 73 *Circuito Transmisor Fisico*..... 107

Figura 74 *Transmisor Inalámbrico*..... 108

Figura 75 *Diagrama de Bloques Metodología Objetivo Específico Numero 4*..... 111

Figura 76 *Algoritmo Funcionamiento Circuito Receptor*..... 112

Figura 77 *Circuito Receptor*..... 113

Figura 78 *Esquema Kicad Circuito Receptor*..... 114

Figura 79 *Modelo 3D hecho en Kicad del Circuito Receptor*..... 115

Figura 80 *Comando para Realizar un Programa Ejecutable Python*..... 117

Figura 81 *Resultado Empaquetamiento del Software*..... 118

Figura 82 *Programa de Simulación del Robot Empaquetado*..... 118

Figura 83 *Consumo de Recursos Software de Simulación*..... 119

Figura 84 *Desplazamiento hacia delante y hacia atrás cámara*..... 120

Figura 85 *Movimientos Izquierda y Derecha Cámara*..... 121

Figura 86 *Rotación Arriba y Abajo Cámara*..... 121

Figura 87 *Rotación Izquierda y Derecha Cámara*..... 122

Figura 88 *Rotación Arriba y Abajo Cámara*..... 123

Figura 89 *Menú de Herramientas Software NJT23*..... 124

Figura 90 *Ventana Informativa Datos Robot NJT23* 124

Figura 91 *Información Longitud Eslabones y Rango Rotaciones* 125

Figura 92 *Ilustraciones Informativa Identificadores de Eslabones y Articulaciones* 126

Figura 93 *Widget Informativo Ubicación Efector Final* 127

Figura 94 *Valor de Rotación de cada Articulación* 127

Figura 95 *Widget Informativo Estado del Software* 128

Figura 96 *Conexión Modulo Receptor al PC* 128

Figura 97 *Verificación Conexión Puerto COM Receptor* 129

Figura 98 *Selección del Puerto COM del Receptor* 130

Figura 99 *Verificación de Conexión con el Receptor* 131

Figura 100 *Ventana de Pruebas del Control Remoto* 132

Figura 101 *Indicación Control Remoto Encendido* 132

Figura 102 *Control Remoto ON* 133

Figura 103 *Apertura Ventana de Pruebas* 134

Figura 104 *Cambio de Estado Mover Joystick Hacia Delante* 135

Figura 105 *Cambio de Estado Mover Joystick hacia atrás* 135

Figura 106 *Cámara Alejada con el Joystick* 136

Figura 107 *Acercamiento de Cámara con Joystick* 136

Figura 108 *Movimiento del Joystick de Rotación Hacia Abajo* 137

Figura 109 *Movimiento Joystick de Rotación Hacia Arriba* 137

Figura 110 *Rotación de la Cámara Hacia Arriba* 138

Figura 111 *Rotación de la Cámara Hacia Abajo* 138

Figura 112 *Prueba de Botón 1 y Selección de Articulación 1* 139

Figura 113 *Prueba Boton2 y Selección Articulación 2*..... 140

Figura 114 *Prueba Botón 3 y Selección Articulación 3*..... 140

Figura 115 *Prueba Botón 4 y Selección de Articulación 4*..... 141

Figura 116 *Prueba Encoder Giro Articulación Base* 142

Figura 117 *Prueba Encoder Giro Articulación Brazo*..... 142

Figura 118 *Giro Encoder Prueba Articulación Antebrazo*..... 143

Figura 119 *Prueba Encoder Articulación Muñeca*..... 143

Lista de Apéndices

Apéndice A *Código Python Multiplicación Matrices* *150*

Introducción

Con la revolución tecnológica actual, se han creado diferentes dispositivos automatizados los cuales tienen la capacidad de realizar trabajos rutinarios y pesados que son difíciles de realizar por los seres humanos. Diferentes empresas internacionales como ABB o FANUC, han diseñado brazos robóticos sofisticados distribuidos por todo el mundo. Colombia es un país fundamental en la región latinoamericana exportando diferentes productos derivados de los recursos naturales a diferentes países. Sin embargo, países como México y Brasil sobrepasan los índices de cantidad de brazos robots manipuladores en sus fábricas comparados con el país colombiano. Dado este contexto surge la pregunta: ¿Cómo contribuir a la capacitación de personal para operar robots manipuladores instalados en las fábricas colombianas? Este trabajo escrito describe el funcionamiento de un software para la programación y análisis de un brazo robot de 4 grados de libertad, dando la posibilidad al usuario de aprender sobre la cinemática directa de un robot, logrando reducir costos en capacitación y posibles daños causados en los procesos de aprendizaje y así contribuir a aumentar el recurso humano colombiano capaz de operar robots manipuladores industriales.

Problema de Investigación

En la actualidad la mayoría de software o programas que se utilizan para la enseñanza de la robótica a los estudiantes son licenciados y de alto costo, por lo tanto, no todas las universidades cuentan con un software integrado con un hardware para brindar los conocimientos específicos en esta área. Como ejemplo ponemos el software de la empresa ABB para la programación de sus robots. Robot Studio es una herramienta de software diseñada por la empresa ABB la cual ofrece múltiples herramientas y controladores virtuales que permiten diseñar trayectorias de sus propios brazos robots para luego llevarlas con lo más mínimo de error posible a la vida real. (ABB, 2023).

El software Robot Studio se encuentra en diferentes plataformas como Windows y Android. En la Figura 1 podemos observar el programa en ejecución de diferentes dispositivos.

Figura 1

Software Robot Estudio Empresa ABB

Fuente. https://new.abb.com/products/robotics/es/robotstudio

Lo que se busca con el proyecto es elaborar un programa que pueda vincular un control remoto en un entorno virtual para la simulación de los movimientos de un robot industrial, dándole a los estudiantes la posibilidad de familiarizarse aún más en el tema de la robótica que tiene poco desarrollo en nuestro país. De esta manera se busca tener un software realizado por

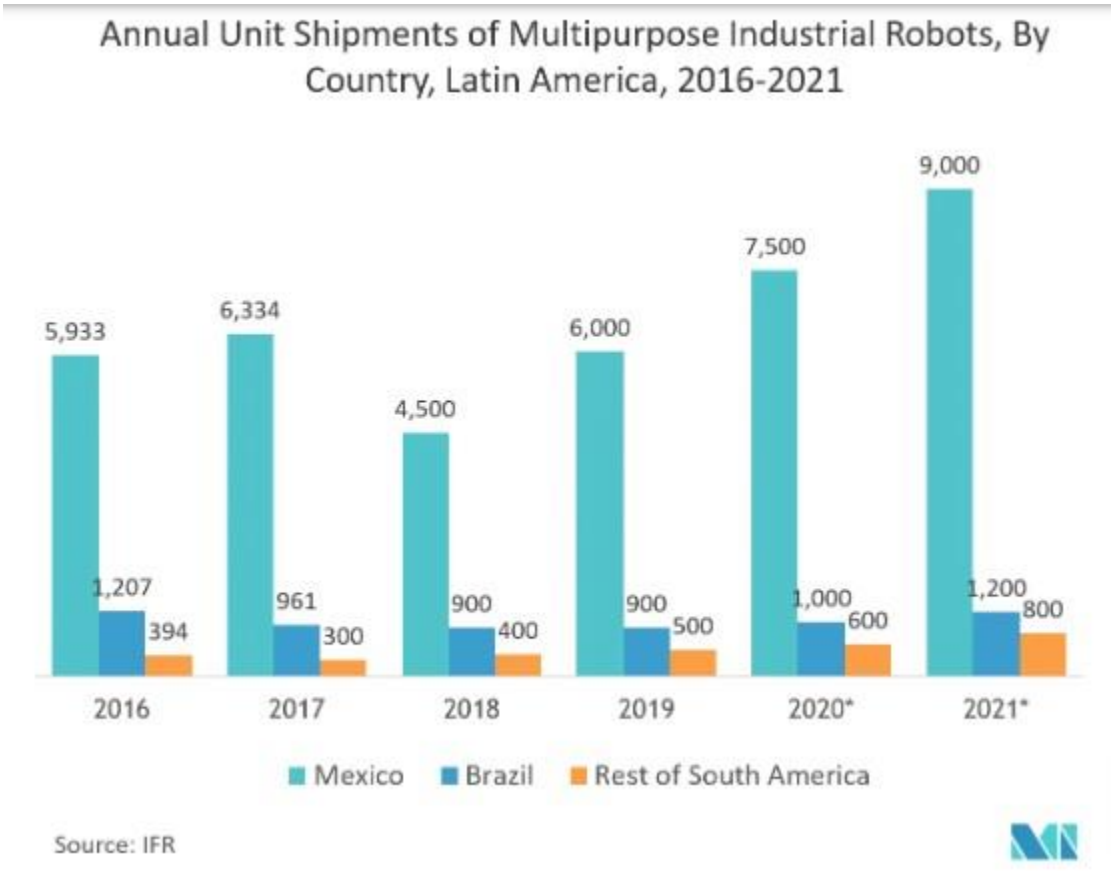
manos colombianas para los colombianos.

Actualmente Colombia es uno de los países en América del sur que menos inversión y desarrollo ha tenido en este campo de la tecnología. Brasil es uno de los más desarrollados en la región, por lo tanto, hemos visto como muy pocas industrias en nuestro país poseen al menos un brazo robot en sus fábrica (Intelligence, 2024), ya sea porque su fabricación es costosa y se deben importar o porque no se encuentra el suficiente personal calificado para la operación de estos dispositivos, aquí es donde juega un papel importante el deseo de diseñar e implementar un software con un control a distancia construido en un entorno virtual de distribución libre.

En la Figura 2 se presenta la cantidad de robots manipuladores distribuidos en México y Brasil comparados con el resto de Latinoamérica, donde es notable que la mayor parte de esta tecnología se concentra en estas dos grandes potencias suramericanas. (Intelligence, 2024), con estos datos cuantitativos podemos concluir que gran parte del desarrollo industrial robótico en el país colombiano se deben a factores socio económicos y educativos. Cabe resaltar las ventajas y desventajas que conlleva implementar esta tecnología en el sector industrial, en donde por un lado tenemos la disminución de costos de producción en las empresas cuando se automatiza una tarea por medio de un robot manipulador, pero también por otro lado se observa menor cantidad de puestos de trabajo en las fabricas en donde gran parte de su porcentaje de producción depende de un brazo robot.

Figura 2

Análisis Cuantitativo Unidades de Robots Industriales Vendidos Multipropósito desde el Año 2016 al 2021

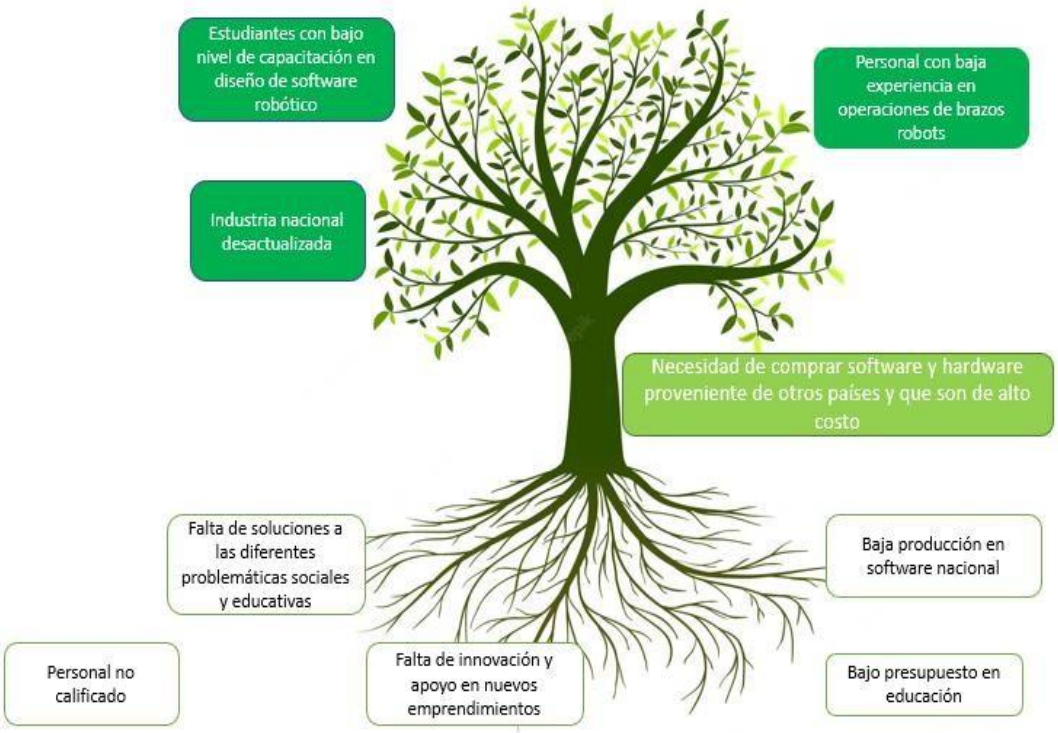


Fuente. https://www.mordorintelligence.com/es/industry-reports/latin-america-warehouse-robotics-market

Colombia es un país que posee la habilidad operativa y las organizaciones educativas para brindar un exponencial desarrollo en temas de automatización empleando brazos robots. Sin embargo, es necesario fortalecer la inversión y la investigación en el tema de diseños de los robots manipuladores. De acuerdo con el texto anterior en la figura 3 se presenta el árbol de causa y efecto de nuestro problema de investigación.

Figura 3

Árbol de Causa y Efecto



Fuente. Elaboración propia.

Dado este contexto surge la pregunta: ¿Cómo contribuir a la capacitación de personal para operar robots manipuladores instalados en las fábricas colombianas? Con este proyecto se busca crear una herramienta que contribuya a la capacitación de personal para operar robots manipuladores industriales.

Objetivos

Objetivo General

Diseñar un prototipo de sistema embebido telemétrico para la simulación de movimientos de un brazo robot modelado en un entorno virtual en un motor grafico de diseño propio.

Objetivo Específicos

Construir un robot manipulador industrial con 4 grados de libertad en un entorno virtual usando software de distribución libre.

Renderizar gráficamente el brazo manipulador industrial modelado, en el software de visualización de movimientos elaborado por el estudiante, utilizando el lenguaje de programación Python y la biblioteca grafica OpenGL.

Implementar un circuito electrónico físico para el control a distancia de un robot manipulador industrial construido en un entorno virtual.

Desarrollar un módulo de comunicación inalámbrico entre el circuito electrónico a distancia y el computador que ejecuta el entorno de simulación donde se encuentra el robot manipulador industrial.

Justificación

Colombia es un país en vía de desarrollo que cuenta con un gran capital humano profesional y una gran cantidad de recursos naturales que pueden llevar a convertirlo en un país altamente desarrollado. Sin embargo, en la actualidad la gran mayoría de productos que se adquieren en el hogar o en la industria son importados.

En el campo de la educación, la gran mayoría de universidades deben adquirir licencias de softwares altamente costosos para poder brindar una orientación más clara en los cursos de las mallas curriculares. Especialmente cuando se abordan temas de robótica se hace uso de Matlab (software matemático de desarrollo científico (MathWorks, 2023)) que al igual que Robot Studio de la empresa ABB cuenta con una licencia altamente costosa (con un costo de una licencia personal 3500USD).

Hoy en día la comunidad de programadores mundiales se ha encargado de trabajar en softwares de código abierto para que personas las cuales quieran realizar un aprendizaje no tenga que sacar dinero de su bolsillo si no que puedan adquirir una copia gratuita de estos programas.

En Colombia el ministerio de tecnologías de la información y las comunicaciones ha tomado la iniciativa de promover el uso de softwares libres incluso en organizaciones gubernamentales de nuestro país (Colombia, 2021)**,** esto con los siguientes propósitos:

- Incentivar la innovación en la administración publica.
- Ahorrar presupuestos en las entidades publicas.
- Evitar la doble contratación de soluciones del estado.
- Promover el desarrollo y uso del software libre.

Dentro del software libre utilizado por el estado colombiano tenemos los siguientes:

- DataSync

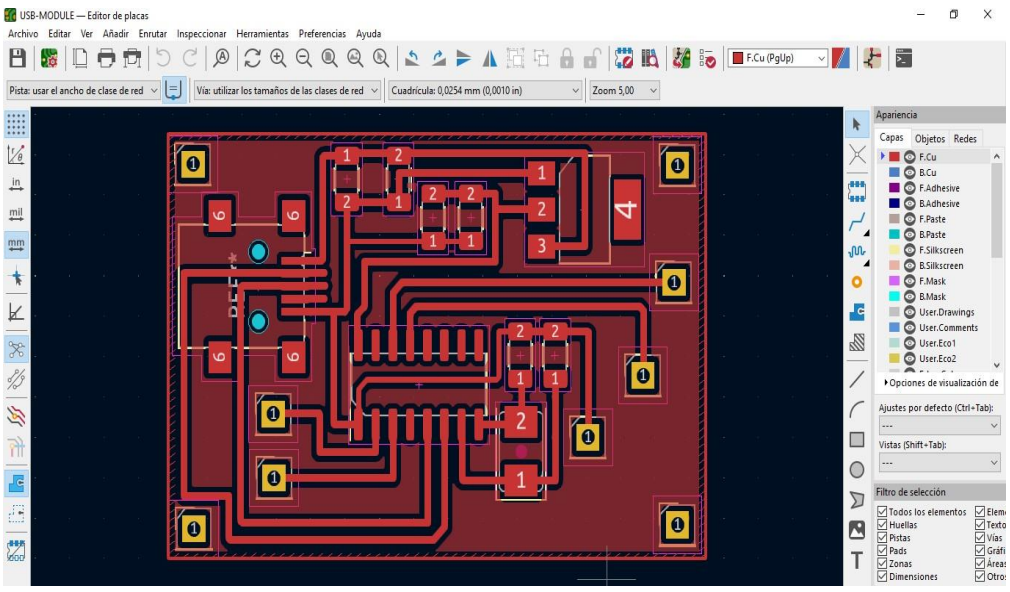
- Pentaho
- MySQL Workbrench
- Python

Podemos observar como ejemplo en nuestro estado colombiano, como el uso de programas bajo licencias de software libre pueden generar impactos positivos como reducir costos en el presupuesto.

Este tipo de programas cuentan con colaboración de desarrolladores a nivel mundial como donaciones y código que se proporciona como mejora y se valida antes de aplicarlo al programa actual. Como ejemplo tenemos el programa de modelado 3D Blender, mientras que para el diseño de circuitos electrónicos tenemos KICAD. En la Figura 4 podemos observar un esquema de circuito electrónico realizado en el software libre KICAD.

Figura 4

Modelado de un Circuito Electrónico Utilizando el Software Libre KICAD



Fuente. Elaboración propia.

Blender por su parte nos permite el diseño de piezas 3D para ingeniería y poder realizar

animaciones, o juegos para otro tipo de artistas digitales 3D, ayudando a la comunidad internacional que no cuenta con los recursos suficientes para comprar programas como maya 3DS, que requiere de licencias altamente costosas (Blender, 2023) de esta forma se logra obtener resultados de bajo costo en los diseños. En la Figura 5 podemos observar la pantalla de inicio de software libre Blender.

Figura 5

Software Libre Blender en su Versión 2.90.1

Fuente. https://docs.blender.org/manual/es/dev/interface/window_system/splash.html

Blender y Kicad siendo ambos dos software de código abierto y de uso gratuito a nivel internacional se han de utilizar en este proyecto. El software Kicad nos será de ayuda para el diseño del circuito Electrónico PCB. Blender por su parte nos ayudará con el modelado 3D del modelo virtual de Robot.

Marco Conceptual y Teórico

En esta sección se dará a conocer los datos técnicos y científicos oficiales de los dispositivos y herramientas necesarias para el diseño del hardware y software del proyecto, teniendo como referencia los primeros antecedentes de los avances en temas de robótica también algunos trabajos relacionados con robótica de código abierto.

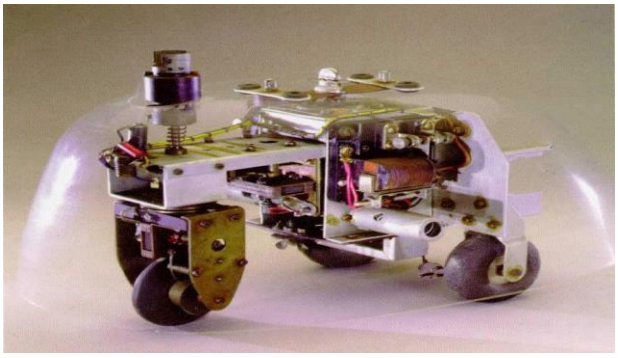
La historia de la robótica data desde la época antigua donde muchos inventores crearon los primeros autómatas. Se dice que en la mitología griega se creó uno de los primeros artefactos. Estos robots fueron estatuas hidráulicas las cuales podían realizar movimientos sencillos o tareas automatizadas como dispensar agua por un cierto periodo de tiempo, estos fueron creadas por el científico e ingeniero Heron de Alejandría (Ortiz, 2023).

Desde la primera guerra mundial gracias a los trabajos de Nikola Tesla, fuimos capaces de crear armas que permitían ser controladas telemétricamente. En los años 1940 cuando ya se tenía un poco más de avance en la rama eléctrica y electrónica el neurólogo William Grey Walter fue un experto en robótica que creo los robots llamados Elmer y Elisa Figura 6, consideradas las primeras maquinas robots de la historia. Los robots funcionaban en base de componentes electromecánicos, eran lentos, pero tenían la capacidad de esquivar obstáculos.(Walter, 1951).

Para su época este tipo de tecnología era realmente futurista ya que un mecanismo autómata electromecánico capaz de esquivar obstáculos por si mismo no era común en el momento. Este proyecto logro crear un hito en la historia de la robótica mundial.

Figura 6

Robot Elmer Diseñado por William Grey Walter



Fuente. http://www.theoldrobots.com/ElmerElsie.html

En 1993 la empresa HONDA, creo los robots de la serie P los cuales ya contaban con una alta gama de microprocesadores, que luego fueron sustituidos por la serie ASIMO. Este fue un robot humanoide cuyo propósito en esa época fue crear un robot capaz de ayudar a las personas con movilidad reducida, pero su objetivo específico era motivar a los estudiantes de las ramas de la electrónica a la creación de dichos autómatas. En la Figura 7 podemos observar al robot ASIMO (Honda, 2023).

Figura 7

Robot Asimo de la empresa Honda

Fuente. https://www.honda.mx/asimo

En la actualidad los robots se encuentran clasificados de la siguiente manera:

- Robots Androides
- Robots Híbridos
- Robots Zoomórficos
- Robots Poli articulados
- Robots Manipuladores.

En el diseño del proyecto nos enfocaremos en los robots manipuladores.

Robot Manipulador

Los robots manipuladores son todos aquellos dispositivos robóticos en donde el operador puede manipular materiales sin tener un contacto físico, es decir, el operador puede manipular cualquier tipo de material por medio del robot. Originalmente se crearon para manipulación de armas explosivas, elementos radioactivos y para algunas exploraciones espaciales. En la actualidad es muy común verlo en las industrias. (Automatica, 2011).

Brazos Mecánicos

Debido a que este tipo de robot se encarga de la manipulación de objetos lo que se busca simular en él es el comportamiento y funcionamiento de un brazo humano. Como ejemplo tenemos el robot industrial. RV-2AJ.

Robot Industrial RV-2AJ

Este es un robot ensamblado y diseñado por la empresa MITSUBISHI. Es un robot industrial manipulador, utilizado en un sin número de aplicaciones como lo son: ensambladoras, pintura, carga y hasta aplicaciones en donde es necesario la visión artificial, también en algunas universidades en todo el mundo. Este modelo es tomado como aprendizaje ya que su programación está disponible en varios softwares de fácil acceso para los estudiantes. Cabe

mencionar que la patente de este robot debido a sus múltiples usos educativos e industriales se encuentra disponible entre varios fabricantes (ELECTRONICS, 2002).

En la figura 8 podemos observar el Robot RV-2AJ de la empresa Mitsubishi, desarrollado por la empresa Festo, gracias al uso de la patente libre de éste.

Figura 8

Brazo Robot RV-2AJ Ensamblado por la Empresa Festo

Fuente. https://docplayer.es/146106503-Universidad-politecnica-salesiana-sede-quito.html

Teniendo en cuenta que el brazo robot que se va a manipular es un modelo virtual, es necesario utilizar una biblioteca grafica por medio de un lenguaje de programación para las operaciones matemáticas, interfaz gráfica y secuencia lógica del programa. Teniendo en cuenta que el diseño del proyecto será basado en software de código abierto A continuación se describe brevemente algunos antecedentes de proyectos de robótica basados en software de código abierto, la biblioteca grafica OpenGL y el lenguaje de programación Python.

Trabajos Relacionados

A continuación, se darán algunos ejemplos de una serie de proyectos y programas de código abierto basados en la simulación robótica esto como referencia de los múltiples esfuerzos de la comunidad de programadores para el desarrollo de software educativo con licencia gratuita.

Controlador Electrónico de Software Libre para la Operación del Brazo Robótico Scorboter_4u

Desarrollado en la universidad técnica de Ambato en ecuador el proyecto se basa en una tarjeta electrónica para el control del Hardware del brazo robótico *scorbot* er_4u, el cual es un robot de código abierto que como objetivo principal tiene promover el aprendizaje, la formación y la educación del control de brazos robóticos industriales, este robot contiene 5 grados de libertad y un peso de tan solo 10KG, basado en un microcontrolador de 32bits lo cual le da una robusta capacidad operativa en su hardware. (Intelitek, 2020). el diseño del controlador electrónico se basa en la familia de microcontroladores *esp8266* y utiliza el protocolo *mqtt* para la comunicación. El proyecto fue desarrollado con el fin de promover la educación de brazos robots en la universidad de Ambato ecuador disminuyendo costos en la compra de hardware patentado para el control de los periféricos del robot. (Chica, 2020)

Diseño y Desarrollo de un Simulador de Código Abierto para un Robot Submarino de Propósito General

Este proyecto desarrollado en la universidad de coruña en España se basa en el desarrollo de un simulador para el control de un robot submarino el cual posee motores, cámara y sonares para la detención de obstáculos, el proyecto fue desarrollado con el software de código abierto Gazebo. (Corella-Solís, 2021) Una de las mejores herramientas de simulación robótica es el software Gazebo que da al usuario la capacidad de construir simuladores para robots con

fenómenos físicos muy realistas (Gazebo, (S. F)). El apartado grafico de este software se realiza bajo la biblioteca OpenGL la cual se explicará en los próximos anexos de este proyecto. El proyecto utiliza el lenguaje de programación de Python en algunos aspectos para la simulación del comportamiento del submarino.

Navegación de Robots Manipuladores en el Entorno de ROS

Desarrollado en la universidad de Zaragoza España este proyecto se basa en la integración de dos robots los cuales se les ha realizado todo el software de control a través del framework de trabajo para la simulación de robots de código abierto ROS. (Gonzalo López Nicolás, 2019). Ros es un entorno de trabajo el cual integra el software necesario para el control y simulación del comportamiento del hardware de un robot, permite crear esquemas controlados de trabajo utilizado por diseñadores y desarrolladores que desean operar programas de código abierto para este tipo de trabajo. (ROS, 2021).

Webots

es un software de código abierto multiplataforma equipado con un completo ambiente de desarrollo brindando a los usuarios la posibilidad de experimentar con la cinemática de varios robots, entre ellos los manipuladores. (Cyberbotics, 2023).

CoppeliaSim

Es un software gratuito el cual contiene una amplia librería de dispositivos robots de marcas tales como Kuka, Intel, Kyocera entre otros dando la capacidad de aprender sobre cinemáticas inversas y directas de grandes variedades de robots. (AG, 2024).

MRPT

Mobile Robot Programming Toolki es una serie de herramientas de código abierto con código fuente en c++, dentro del proyecto se encuentran herramientas de simulación de Kinect,

sin embargo, una de las herramientas más importantes en el software de simulación del algoritmo de Denavit Hartenberg. (MRPT, 2024).

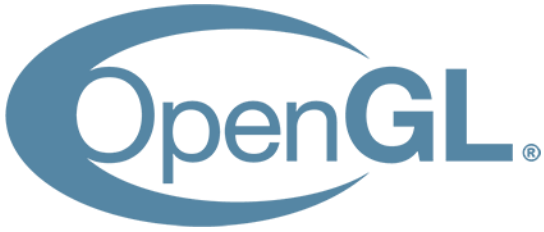
OpenGL

Es una librería grafica 2D y 3D multilenguaje y multiplataforma la cual nos ayuda a la renderización de gráficos complejos por medio de los diferentes métodos y funciones que nos provee. OpenGL es capaz de alojar los gráficos en la GPU o memoria RAM de las computadoras permitiendo así que las operaciones matemáticas y compilación de código se lleve a cabo en la CPU y la visualización de gráficas y renderización en la GPU permitiendo hacer aplicaciones 2D y 3D fluidas. (Group, 2022).

En la figura 9 podemos observar el logo de esta destacada librería el cual ha permanecido durante los últimos 5 años.

Figura 9

Logo Biblioteca OpenGL

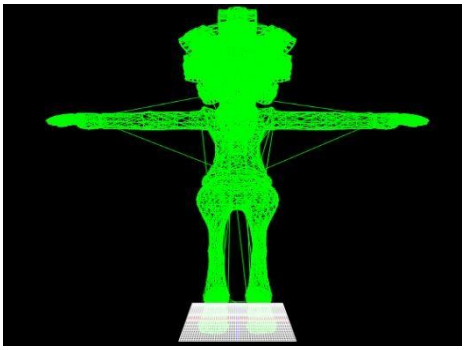


Fuente. https://www.opengl.org/

Actualmente OpenGL se encuentra a cargo del grupo Kronos y gran variedad de softwares industriales y educativos corren bajo esta API, entre ellos tenemos: Maya 3DS, Fbx Viewer AutoDesk, Blender, DOOM3, Hitman, y algunas versiones de Unreal Engine. También cuenta con un amplio soporte en lenguajes de programación C++ y Python. En la figura 10 podemos ver la renderización de gráficos 3D por malla usando OpenGL y Python.

Figura 10

Renderización de Gráficos 3D con OpenGL



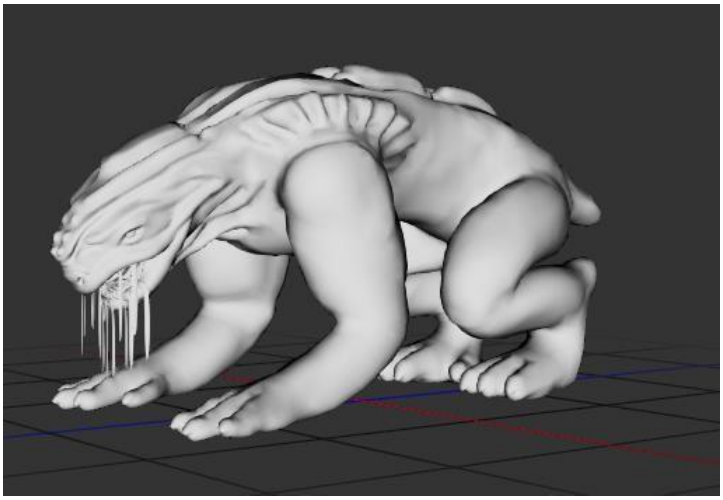
Fuente. Elaboración propia

OpenGL nos permite la visualización de primitivas 3D y 2D como los son polígonos, triángulos, círculos, y líneas. De esta manera y mediante algoritmos podemos representar una serie de modelos 3D en nuestra pantalla grafica. OpenGL se basa en algebra matricial para su funcionamiento, por lo tanto, la representación de vectores o puntos en el espacio se hace mediante vectores que pueden ser *Vec3 y Vec4*. Un vector *Vec3* estaría construido por 3 componentes Vec3(x, y, z). Un vector de 4 componentes se le añadirá el componente w al final Vec4(x, y, z, w). En la parte matemática este componente se usa para realizar operaciones matriciales entre sí.

La renderización de gráficos en OpenGL puede ser tan optima que nos permite visualizar objetos simulando luces y sombras para tener efectos más realistas. Como podemos observar en la figura 11 se ha cargado un modelo 3D complejo con sombras y luces para visualizar un personaje ficticio de Alíen.

Figura 11

Renderización de Gráficos 3D con OpenGL Programa F3d



Fuente. Elaboración propia

Se puede realizar transformaciones geométricas a los modelos 3D de nuestro programa por medio de matrices de transformaciones las cuales pueden ser de 3 tipos: Matrices de escala, matrices de rotación y matrices de traslación. Los modelos gráficos 3D en un entorno virtual están compuestos por vectores como hemos descrito en párrafos anteriores, las matrices de transformación nos ayudan a realizar operaciones a los vectores para dar movimiento según lo necesario, traslación o rotación. En las siguientes secciones se describe la operación y resultado de cada una de las matrices.

Matrices de Escala

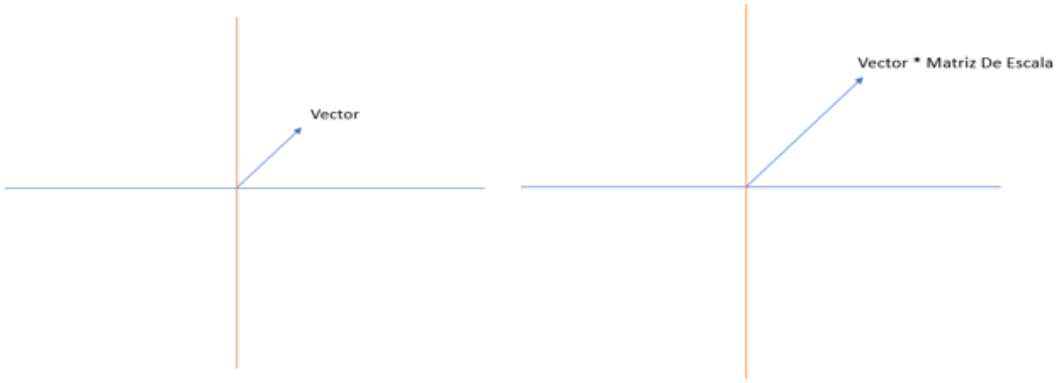
Una matriz de escala 4x4 es una matriz en cuya diagonal se encuentra un factor de escala que permite ajustar el tamaño de un vector o un modelo 3D en el mundo digital. Está representada de la siguiente manera.

$$Matriz\ de\ escala = [sx0000sy0000sz00001]$$

Por lo tanto, al multiplicar un vector de cuatro componentes por nuestra matriz de escala tendremos como resultado un nuevo vector en el espacio, con un tamaño que puede ser mayor o menordependiendo de nuestro valor de escala. En la figura 12 podemos ver un vector por una matriz de escalamiento.

Figura 12

Escalamiento de un Vector



Fuente. Elaboración propia

Matriz de Rotación

Es una matriz 4x4 la cual nos permite rotar un vector o un punto en el espacio. Hay una matriz de rotación para cada eje. Es posible multiplicar las 3 matrices de rotación juntas para dar por resultado una sola matriz (Bernard Kolman, 2006)**.**

$$Matriz\ De\ Rotacion\ En\ X = [10000\cos\theta - \sin\theta 00\sin\theta\cos\theta 00001]$$

$$Matriz\ De\ Rotacion\ En\ Y = [\cos\theta 0\sin\theta 00100 - \sin\theta 0\cos\theta 00001]$$

$$Matriz\ De\ Rotacion\ En\ Z = [\cos\theta - \sin\theta 00\sin\theta\cos\theta 0000100001]$$

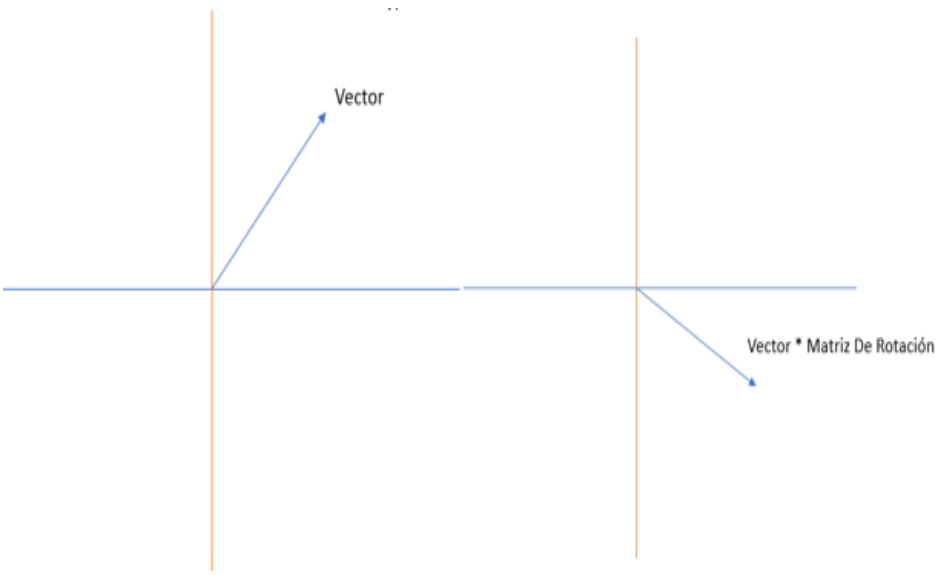
Sin embargo, en aplicaciones graficas más complejas se utilizan matrices de quaterniones en lugar en matrices de rotaciones Euler para evitar problemas de procesamiento y movimientos de vectores en un espacio 3D.

Los quaterniones utilizan los números complejos e identidades trigonométricas para la rotación de modelos, simulación de objetos en espacio 3D. sin embargo para el alcance de este proyecto es suficiente trabajar con matrices de rotaciones Euler como se han descrito previamente.

En la figura 13 podemos observar la rotación de un vector en un plano 2D por medio de estas matrices, en este caso se ha utilizado solo un plano 2D en donde tan solo existen los planos X y Y.

Figura 13

Rotación de un Vector



Fuente. Elaboración propia

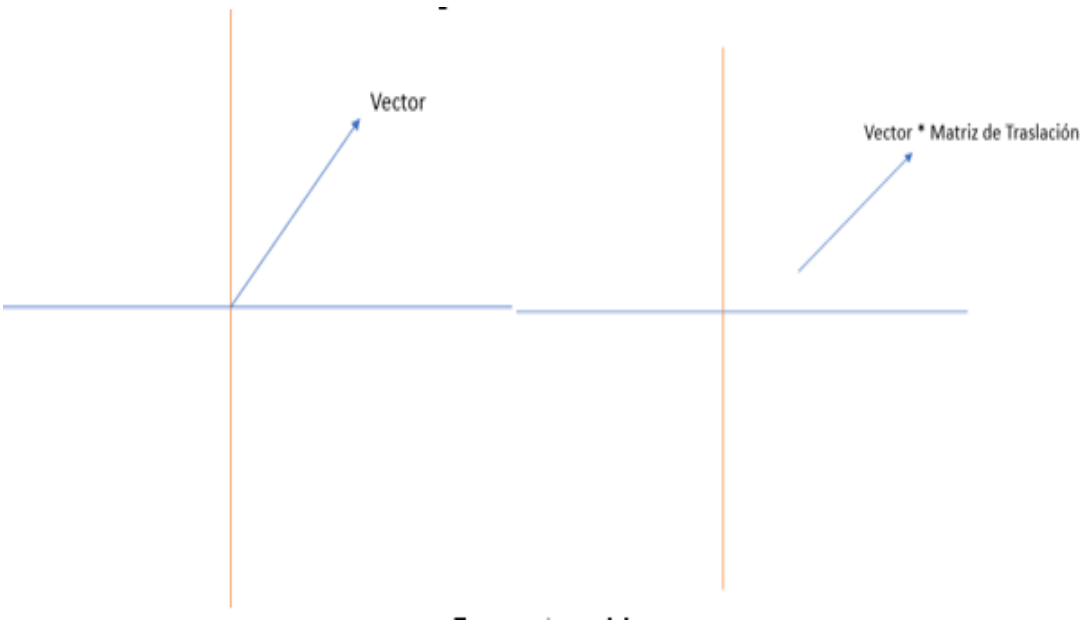
Matriz de Traslación

Es una matriz 4x4 que permite mover o desplazar un vector de un lugar a otro (Bernard Kolman, 2006). En la figura 14 observamos como movemos la posición de un vector por medio de esta matriz.

$$Matriz\ de\ traslacion = [100x010y001z0001]$$

Figura 14

Traslación de un Vector



Fuente. Elaboración propia

Hasta el momento se ha descrito la herramienta grafica OpenGL la cual nos ayudara a visualizar el modelo del brazo robot en el entorno virtual y cómo funcionan las matrices de transformación junto con los vectores de los modelos 3D para realizar operaciones de traslación, rotación y escalamiento. Sin embargo, todas estas bases no son posible realizarse en un software sino es con la ayuda de un lenguaje de programación. En la próxima sección se hablará un poco sobre el lenguaje de programación Python. El cual se utilizará para el diseño completo del software.

Python

Python es un lenguaje de programación multiparadigma de los más utilizados en la actualidad. Se usa para realizar Software de escritorio, Aplicaciones Web, Machine Learning,

Big data e inteligencia artificial. Para el diseño del proyecto se usa en conjunto con las bibliotecas de interfaces graficas Qt Designer y PYQT5.

Estas dos últimas bibliotecas nos dan una variedad de widgets como botones, barras de progreso, Gráficos, Scroll bar, barras de menús entre otros para la elaboración de programas robustos de escritorios. En la figura 15 se puede observar el logo de Python junto con el de Qtdesigner.

Figura 15

Python y Qtdesigner

Fuente. Elaboración propia

Python, OpenGL y la matemática matricial serán las herramientas bases para el modelado del robot y diseño del software gráfico que funcionará en el sistema operativo Windows. Sin embargo, para realizar los movimientos de las articulaciones del robot manipulador se hará de manera remota por medio de un control Hardware. En la próxima sección se describirá los componentes y protocolos de comunicación a utilizar para el diseño del control inalámbrico.

ESP8266

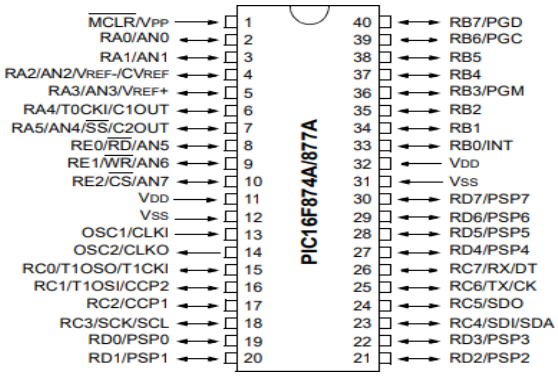
Es un chip de bajo consumo y bajo costo que posee una CPU con una velocidad de procesamiento muy rápida llegando hasta los 80MHz. Fue diseñado por la empresa CHINA Espressif y cuenta con la capacidad de conectarse a una red Wifi. Su Sucesora es la ESP32 que es capaz de comunicarse vía bluetooth. El ESP8266 además cuenta con Interfaz SPI e I2C y con un solo canal ADC de 10 Bits. Éste se utilizará junto con protocolos ESPNOW para la elaboración del control Telemétrico. (Espressif, 2022).

PIC16F877A

Este es un microcontrolador diseñado por la empresa microchip que al igual que el ESP8266 tambien cuenta con conversores ADC de 10bits, pero posee 33 pines de entrada y salida, 8 canales ADC de 10 bits, memoria EEPROM de 256 Bytes, Interfaz de comunicación SPI, I2C, 2 Timers de 8Bits y 1 Timer de 16Bits. En la figura 16 podemos observar el digrama de pines de este microcontrolador (Microchip, 2019).

Figura 16

Diagrama de Pines PIC16F877A



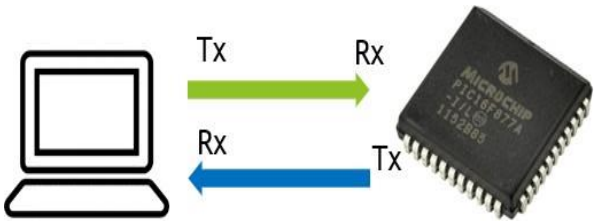
Fuente. https://www.microchip.com/en-us/product/PIC16F877A

Interfaz de Comunicación Serial UART

Es un tipo de comunicación de 8Bits, esto quiere decir que podemos enviar hasta 256 caracteres diferentes por medio de un solo bus de datos. Este tipo de comunicación es bidireccional, de manera que podemos enviar datos y recibir al mismo tiempo. Se usa para comunicar computadores con otros periféricos como los microcontroladores o también en envió y recepción de datos entre estos microprocesadores. En la Figura 17 se muestra el diagrama de conexión entre un microcontrolador y un computador.

Figura 17

Conexión PC - Microcontrolador

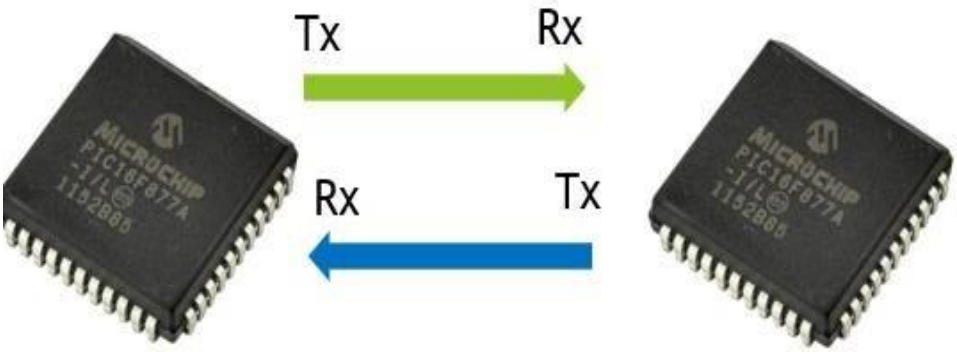


Fuente. Elaboración propia

Cada uno de los periféricos cuenta con dos pines para el envío y recepción de datos. Para enviar se utiliza el pin Tx y para recibir se utiliza el Pin Rx. También podemos enviar y recibir información entre microcontroladores (Microchip, 2019). En la figura 18 mostramos un esquema de conexión entre Mcu y Mcu.

Figura 18

Conexión Microcontrolador - Microcontrolador



Fuente. Elaboración propia

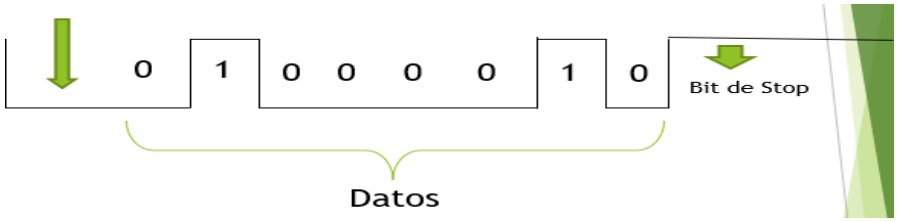
Pasos de funcionamiento:

- Se genera un bit indicando el inicio de transmisión.
- Se envía una cadena de 8bits indicando el dato.
- Se genera un bit de stop indicando el final de la transmisión.

En la figura 19 podemos observar como viaja una señal con un dato desde el transmisoral receptor utilizando el protocolo de comunicación UART.

Figura 19

Grafica de Datos Protocolo Serial UART



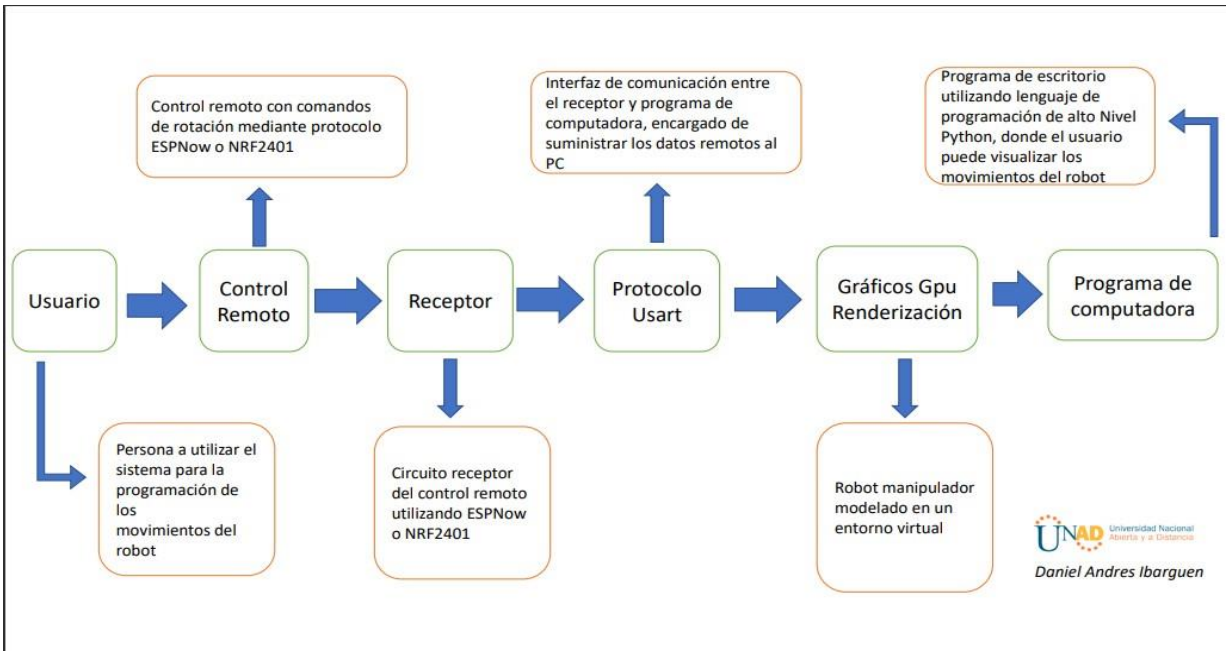
Fuente. Elaboración propia

Podemos observar que la cadena de datos corresponde Al numero binario 0b : 01000010 que en Hexadecimal es Hex : 0X42.

Para el proyecto se utiliza este tipo de interfaz de comunicación para recibir los datos de entrada y convertidores ADC a traves del PIC16F877A y ser enviados via puerto Serial al ESP8266 para finalizar siendo enviados a la interfaz de forma inalambrica como se puede observar en la Figura 20.

Figura 20

Diagrama de Bloques del Sistema Utilizando Sistema de Comunicación ESPNOW



Fuente. Elaboración propia

En la Figura 21 podemos observar el diagrama gráfico del proyecto en donde se describe por medio de simbolos el funcionamiento del sistema.

Figura 21

Diagrama Grafico

Fuente. Elaboración propia

Metodología

En este capítulo describiremos la metodología utilizada para la elaboración del proyecto. Se dará a conocer el alcance de investigación, el enfoque, instrumentos de investigación y el cronograma de actividades de este proyecto.

Enfoque de Investigación

En el enfoque de investigación del proyecto es cuantitativo debido a que se llevara una serie de pasos para poder llegar a la finalidad de la investigación y no se puede eludir alguno de ellos. Como ejemplo no podemos realizar el modelamiento de un brazo robot, si primero no se tiene un boceto, prototipo o bosquejo de las forma física o virtual del robot y el número de articulaciones por el medio del algoritmo de Denavvit Hartenberg.

El doctor Roberto Hernández sampieri en el libro de metodología de investigación dice *"*un proyecto de investigación tiene enfoque cuantitativo cuando representa un conjunto de procesos, es secuencial y probatorio"**.** (Roberto Hernandez Sampieri, 2010)**.**

Alcance de Investigación

El Alcance de la investigación será descriptiva y correlacional, debido a que se analizará la relación entre las diferentes variables y métodos matemáticos para las rotaciones de cada una de las partes del robot. Como se ha descrito anteriormente en el marco teórico OpenGL y el algoritmo de D*enavit Hartenberg* se basa en matemática matricial para la representación de vectores, rotación, traslación y escalamiento, para la formación de objetos en 3D y el modelado de brazos robots. Por lo tanto, es necesario que la investigación sobre gráficos por computadora utilizando la biblioteca grafica OpenGL y matemática matricial tenga un alto grado de compresión.

El doctor Carlos Fernández collado menciona en su libro colaborativo en conjunto con

Roberto Sampieri. “Los estudios descriptivos buscan especificar las propiedades, las características y los perfiles de personas, grupos, comunidades, procesos, objetos o cualquier otro fenómeno que se someta a un análisis”. (Roberto Hernandez Sampieri, 2010)

Diseño de Investigación

Profundizando poco en el diseño experimental, se debe analizar el comportamiento del envió inalámbrico de variables como lo son: variables de rotación del robot y cámara, variables de control y elección mediante una interfaz serial y otra interfaz inalámbrica. Se debe investigar y comprobar mediante código y hardware la idea que se plantea al realizar un control inalámbrico.

Instrumentos de Investigación

Para llevar a cabo este proyecto se utilizaron los siguientes instrumentos de investigación:

- Datasheets y documentos oficiales de los componentes electrónicos utilizados.
- Documentación oficial de OpenGL.
- Foros de investigación de software libre.
- Guías oficiales de desarrollos de interfaces graficas Pyqt5.
- Libros de modelamiento de robots.

Todos los instrumentos aquí mencionados se encuentran en las referencias bibliográficas de este documento.

Cronograma de Actividades

Tabla 1

Cronograma de Actividades del Proyecto

Actividad	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5	Mes 6
Realizar el modelado de un brazo robot industrial en un entorno virtual de software libre.	X					
Diseñar un algoritmo y un software para representar el brazo robot modelado en un entorno virtual, en el futuro programa propio de computadora		x				
Implementar un circuito de control inalámbrico para los movimientos a distancia del brazo robot usando PIC16F877A			x			

Actividad	Mes 1	Mes 2	Mes 3	Mes 4	Mes 5	Mes 6
Realizar un circuito receptor usando un ESP8266				x		
Diseñar un algoritmo utilizando el protocolo UART para obtener los datos del control remoto en el computador					x	
Pruebas finales de implementación del proyecto						x

Nota En la tabla 1 podemos observar el orden y el tiempo en el que se desarrollaran las actividades del proyecto. *Fuente.* Elaboración propia

Construcción del Brazo Robot en el Software de Distribución Libre Blender

En esta sección se explicará la metodología utilizada para la creación del brazo robot, utilizando el entorno de modelamiento 3D Blender. Este software nos brinda una serie de técnicas y procesamiento de datos gráficos para la construcción de modelos y escenas 3D. El robot cuenta con 4 grados de libertad y 5 eslabones el cual se modelará usando los pasos propuestos por Denavit – Hartenberg los cuales explicaremos en este capítulo.

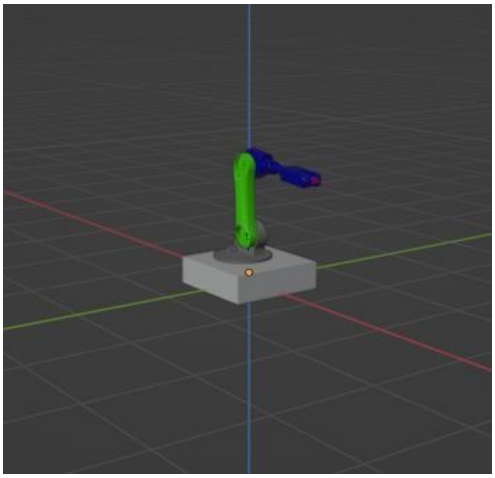
Blender es un software de diseño 3D de código abierto. para utilizarlo no se requiere de licencia. En el podemos realizar esculpido, coloreado, escalado y diferentes transformaciones de vértices para llegar a una escena o diseño final. Para nuestro caso en donde no tenemos la suficiente experiencia en modelado 3D hemos modificado y diseñado partes de acuerdo con modelos mecánicos de ejemplos de brazos robot comerciales.

Hemos denominado el modelo del robot como NJT-23 con un total 4 grados de libertad. En la figura 22 podemos ver el diseño de nuestro brazo robot. NJT23 es un robot articulado, que cuenta con partes que pueden realizar movimientos rotatorios en los ejes X, Y y Z. Normalmente se definen los grados de libertad del robot en función de los movimientos de las articulaciones.

El Robot NJT23 es el primer diseño que se utilizara para la simulación de los movimiento o la cinemática directa, la cual funcionara basada en los algoritmos de denavit hartenberg y la matemáticas matricial la cual se mostraran en secciones más adelante.

Figura 22

Diseño de Brazo Robot con 4 Grados de Libertad Utilizando el Software Blender

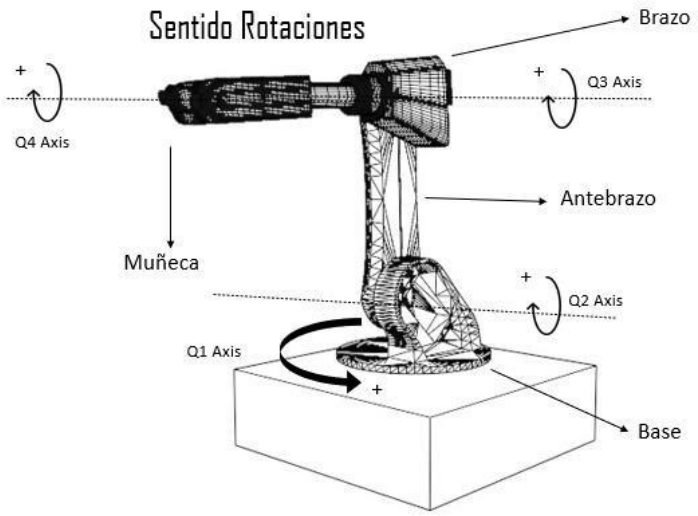


Fuente. Elaboración propia

En la figura 23 podemos observar el despiece del robot construido en Blender.

Figura 23

Despiece del Robot NJT-23



Fuente. Elaboración propia

Tabla 2

Tabla de Longitudes y Alcance en Grados de las Articulaciones

Item		Unidad	Especificación
Longitud del brazo robot NJT23	Base	**Cm**	40
	Antebrazo	**Cm**	100
	Brazo	**Cm**	100
	Muñeca	**Cm**	24
Rangos de operación	Base	**Grados**	240° (120° a -120°)
	Antebrazo	**Grados**	140° (90° a -50°)
	Brazo	**Grados**	125° (90° a -35°)
	Muñeca	**Grados**	180° (90° a -90°)

Nota: En la tabla anterior se puede observar los tamaños en centímetros y los rangos de rotaciones de las articulaciones. *Fuente*. Elaboración propia.

Tabla 3

Identificadores de Eslabones y Articulaciones

Identificador del eslabón	Base	**ID**	E0
	Antebrazo		E1
	Brazo		E2
	Muñeca		E3
Variable articular	Base	**ID**	q0
	Antebrazo		q1
	Brazo		q2
	Muñeca		q3

Nota. Los identificadores corresponden a la variable que se utilizara para cada articulación y eslabón para el modelado del algoritmo de Denavit – Hartenberg. *Fuente*. Elaboración propia

Sistema de partes

Describiremos el funcionamiento y movimiento de cada una de las articulaciones y eslabones del robot

Base

La base es la estructura que nos permite realizar rotaciones sobre un eje vertical en todo el brazo robot exactamente en el Angulo ubicado en el eje Z, en la Figura 24 se puede apreciar el modelo individual de la base que soporta el resto de las partes del brazo robot.

Figura 24

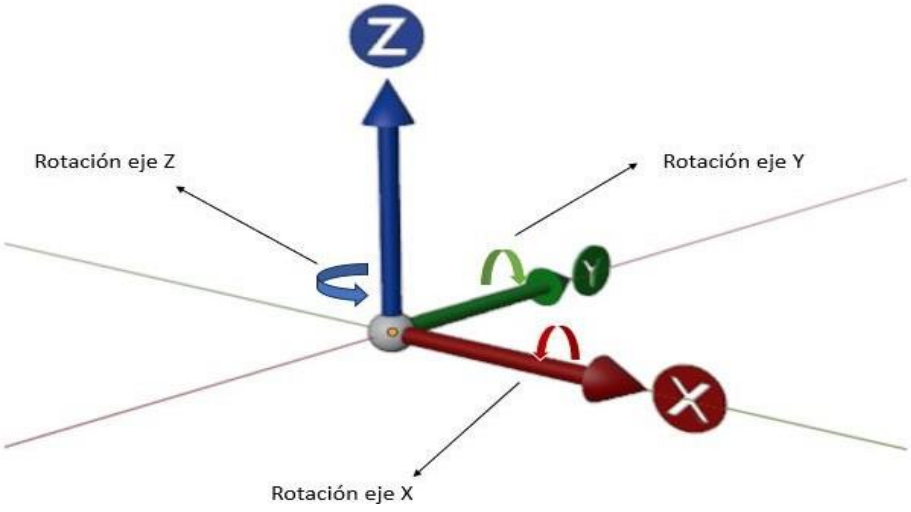
Modelo 3D Base Robot NJT23

Fuente. Elaboración propia

En la figura 25 se presenta el sistema de referencia base del brazo robot, en donde se puede observar las orientaciones de los ejes de rotación X, Y, Z. La rotación de esta articulación afectara todas las demás articulaciones en dicho Angulo. El eje de rotación de esta articulación se encuentra en su zona central, circunferencia base, como se observa en la figura 26.

Figura 25

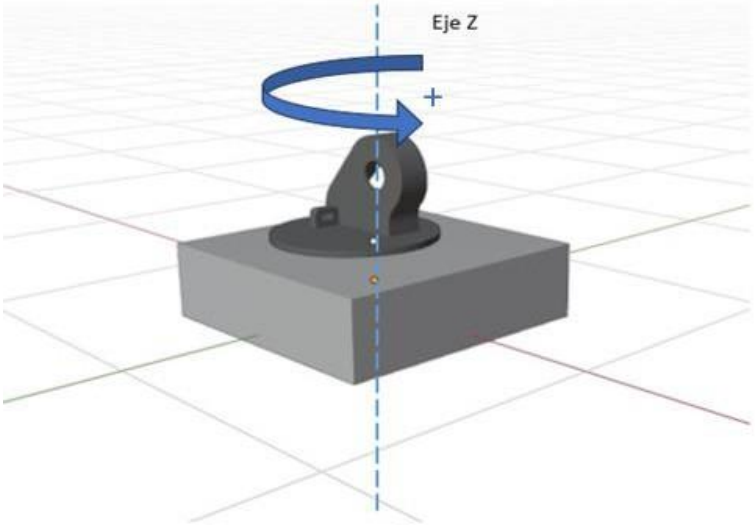
Sistema de Referencia Base del Brazo Robot



Fuente. Elaboración propia

Figura 26

Eje de Rotación de la Base



Fuente. Elaboración propia

Antebrazo Brazo Robot NJT-23

Figura 27

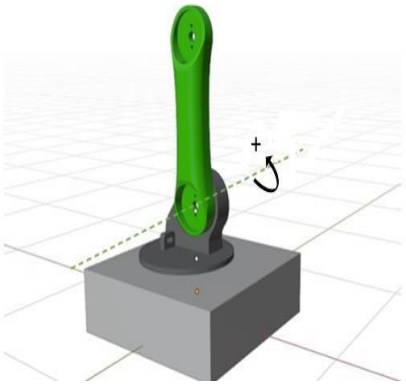
Antebrazo del Robot NJT23

Fuente. Elaboración propia.

El antebrazo se encuentra anclado a la base en su eje central. Ésta se encarga de hacer rotaciones en el eje Y en la figura 27 se muestra el modelo sólido y de malla del antebrazo, en la figura 28 podemos observar cómo está anclada a la base.

Figura 28

Eje de Rotación del Antebrazo del Robot NJT 23

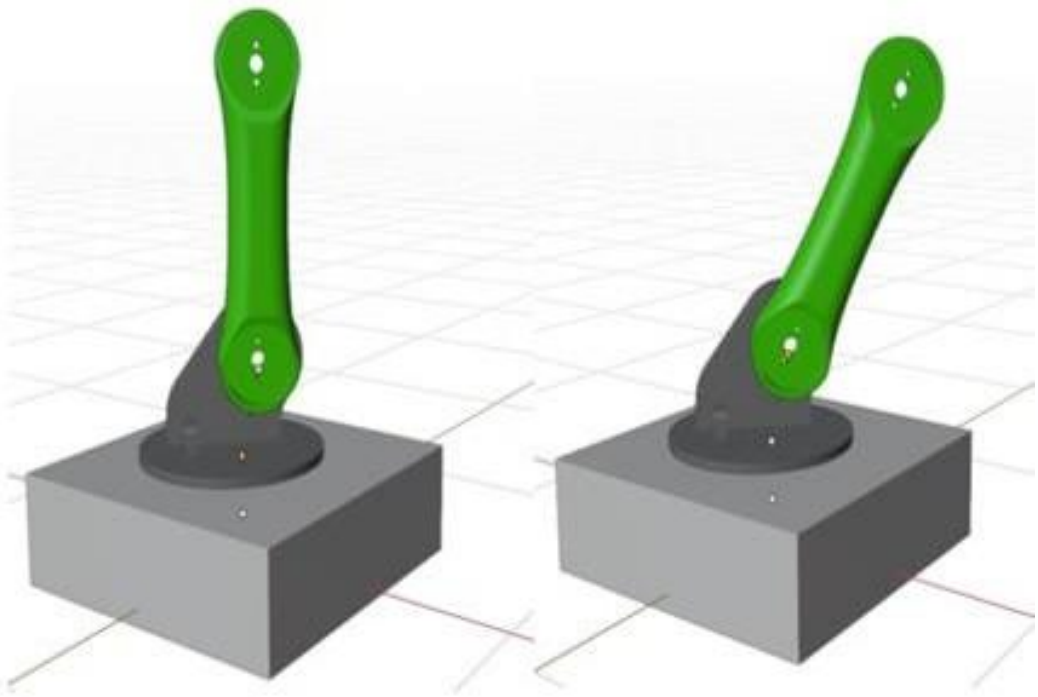


Fuente. Elaboración propia.

En la figura 29 podemos observar las posibles rotaciones de las dos primeras articulaciones. En la imagen de la izquierda se ha rotado 45 grados en el eje Z la articulación base y a la derecha se ha rotado -30° grados en el eje Y el antebrazo del brazo robot NJT-23

Figura 29

Rotación Base y Antebrazo



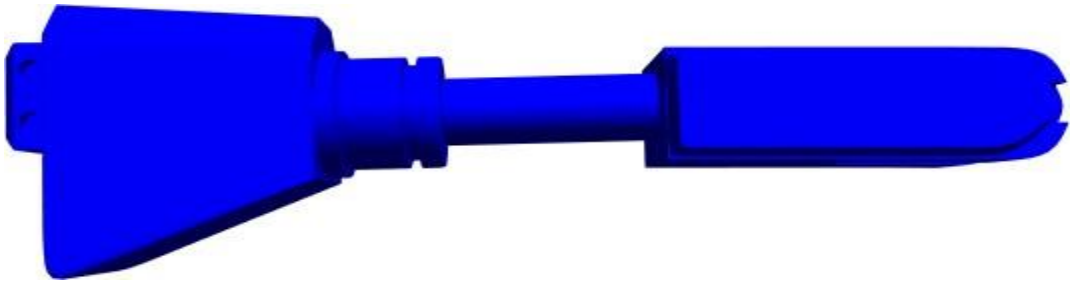
Fuente. Elaboración propia.

Brazo del Robot NJT-23

El brazo del robot NJT23, figura 30 se encuentra anclado a la articulación anterior. Su rotación independiente será en el eje Y. Al estar anclado a la articulación previa compartirá sus valores de rotación partiendo desde su eje central que hemos ubicado como podemos observar en la figura 31.

Figura 30

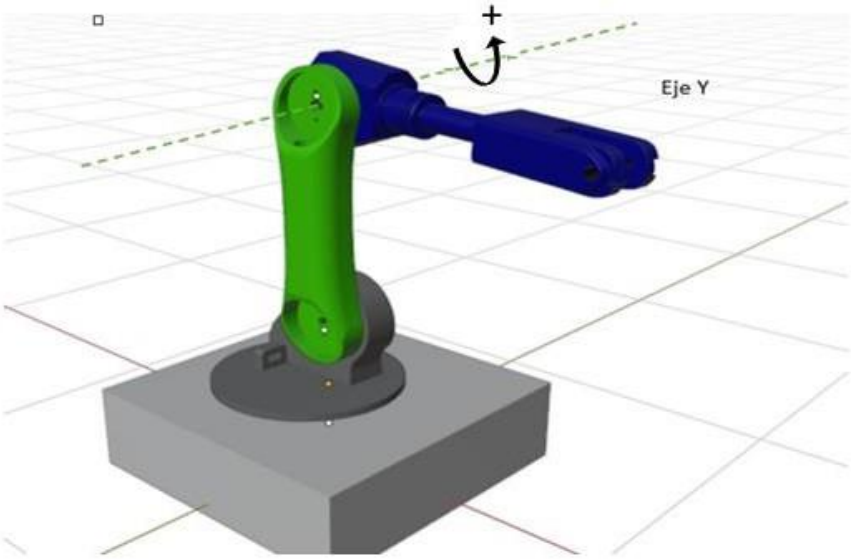
Brazo Robot NJT-23



Fuente. Elaboración propia.

Figura 31

Ubicación el Eje Central Articulación 3

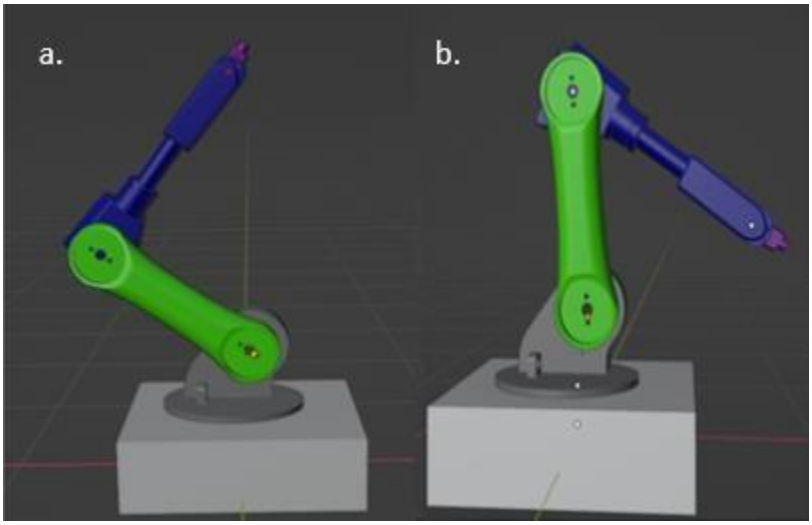


Fuente. Elaboración propia

En la figura 32 podemos observar las rotaciones dependientes e independientes de esta articulación. En la imagen del lado izquierdo tenemos una rotación de la articulación anterior **"Antebrazo"** con un valor de 55° grados. En la imagen de la derecha hemos rotado la articulación actual **"brazo"** de manera independiente con un valor de **-45°** grados.

Figura 32

Rotación Dependiente e Independiente



Nota. a. rotación dependiente b. rotación independiente *Fuente.* Elaboración propia

Muñeca del Brazo Robot

La muñeca es una de las partes más importantes del Brazo robot, siendo esta la parte en donde se sitúa la herramienta que podría utilizar el brazo robot. En este caso podría llamarse el efector final. En los próximos párrafos daremos a conocer el algoritmo de Denavit Hartenberg para situar el efector final en una posición especifica. En la figura 33 se puede ver el modelo 3D de esta parte del brazo Robot.

Figura 33

Muñeca o Brida del Brazo Robot NJT-23

Fuente. Elaboración propia

Esta articulación se encuentra anclada a la parte anterior (brazo), como se puede ver en el árbol que representa la Jerarquía de cada una de las articulaciones figura 34.

Figura 34

Jerarquía Articulaciones

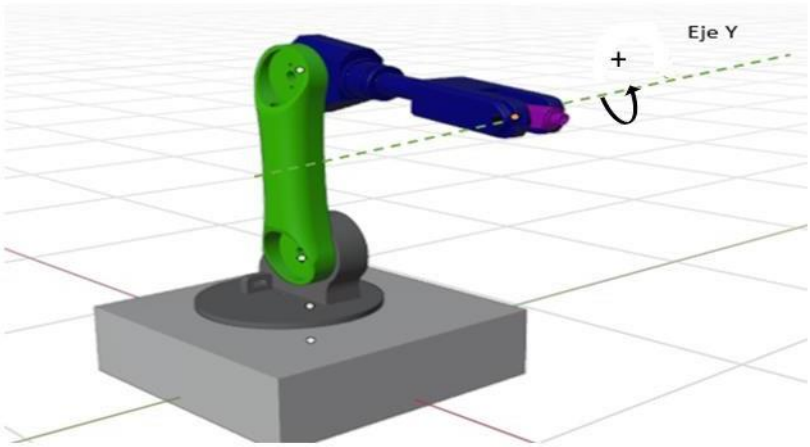
Fuente. Elaboración propia

Analizando la figura anterior se puede finalmente observar que la articulación 1 - Cadera es la articulación padre de las otras articulaciones. Podemos observar que la articulación 3 - Brazo es dominada por la articulación anterior 2 - Hombro **y** a su vez es padre de la articulación final: muñeca.

Esta articulación tendrá la libertad de moverse de forma independiente en el eje Y como se ha descrito en la figura 25. El posicionamiento del eje de esta articulación se ha situado como podemos observar en la figura 35.

Figura 35

Posicionamiento Eje de Rotación Aticulación 4

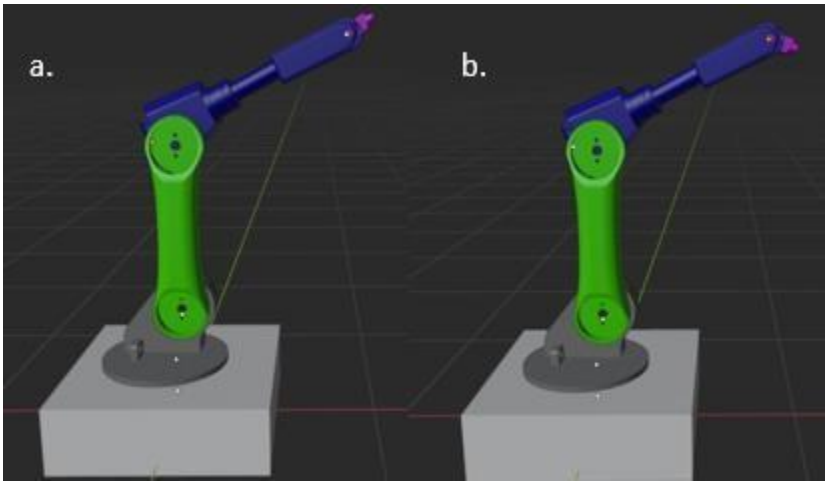


Fuente. Elaboración propia.

En la figura 36 se puede observar las rotaciones dependientes e independientes de esta articulación. Se ha rotado el *"brazo"* con un valor de 30° grados y se ha rotado la *"muñeca"* de manera independiente con un valor de -60° grados.

Figura 36

Rotación Dependiente e Independiente Articulación Muñeca



Nota. a. Rotación de articulación anterior brazo -30 grado. B. rotación de muñeca -60 grados

Fuente. Elaboración propia.

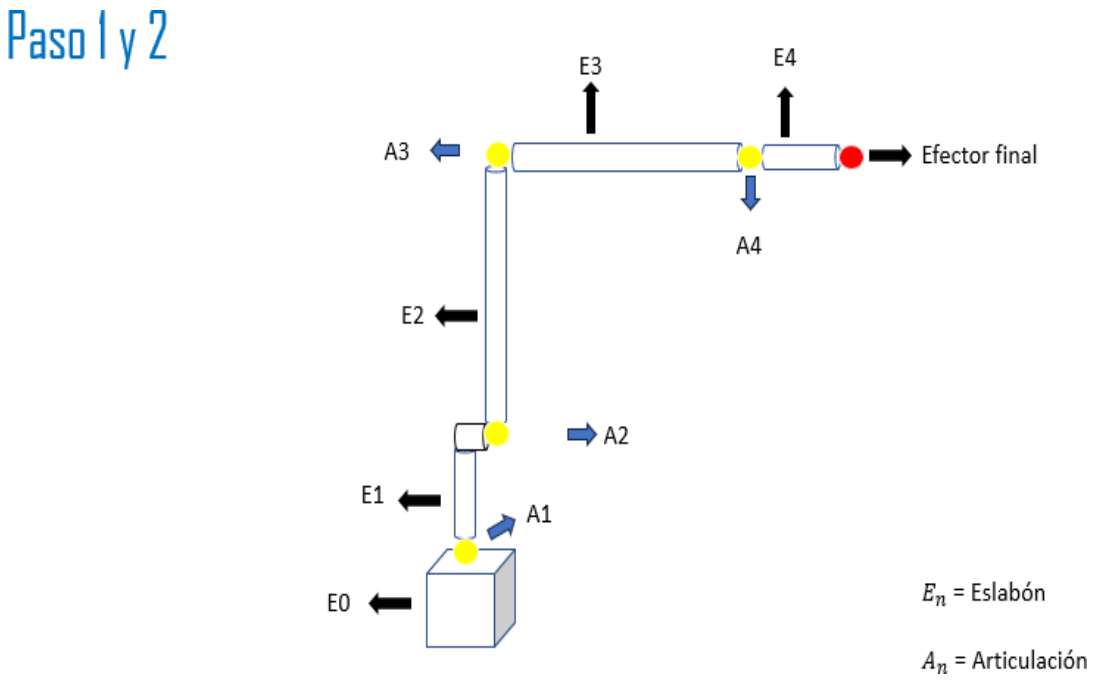
Algoritmo de Denavit-Hartenberg

En el año 1955 Denavit y Hartenberg propusieron un método matricial que permite establecer la posición del efector final o de una cadena mediante una serie de pasos y operaciones geométricas, permitiendo definir la cinemática directa de un robot**.** (Antonio Barrientos, 2007) Los pasos del algoritmo son los siguientes:

1. Identificar los eslabones comenzando por 0 donde el eslabón 1 será el primer eslabónmóvil de la estructura robótica
2. Identificar cada articulación comenzando con 1 correspondiente al primer grado de libertady terminando en n figura 37

Figura 37

Identificación de Eslabones y Articulaciones



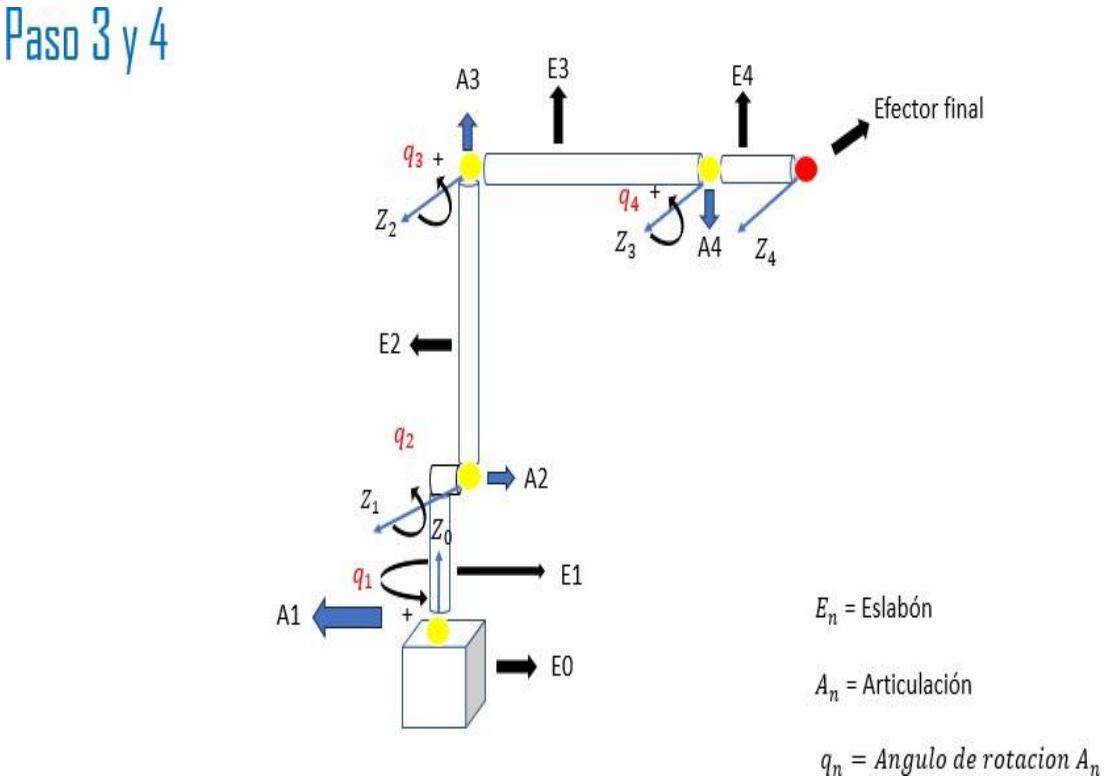
Fuente. Elaboración propia.

3.Localizar el eje de cada articulación. Si la articulación es rotativa, el eje será su propioese de giro

4.Para i de 0 a n-1 situar el eje Zi sobre el eje de la articulación i + 1 Figura 38

Figura 38

Paso 3 y 4

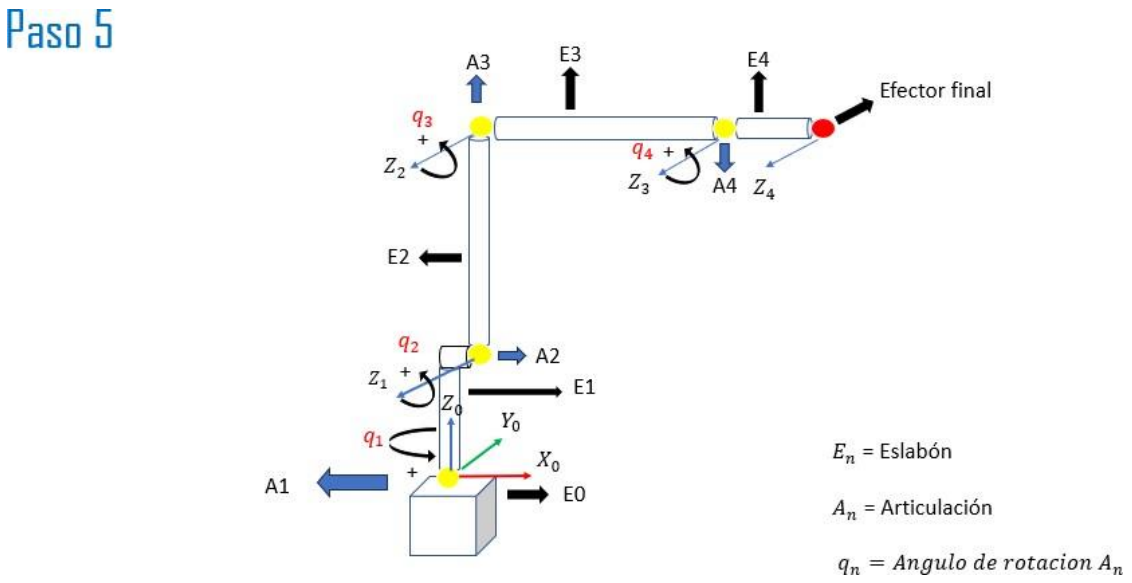


Fuente. Elaboración propia.

5.Situar el origen del sistema de la base $S0$ en cualquier punto del eje $Z0$. Los ejes $X0$ y $Y0$ se situarán de modo que formen un sistema dextrógiro con $Z0$. Figura 39

Figura 39

Paso 5

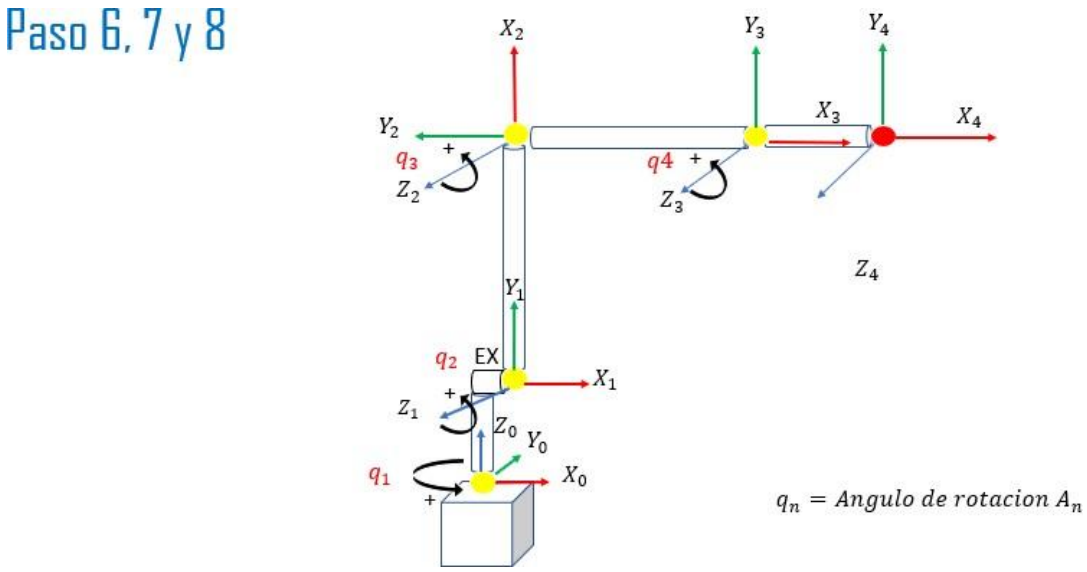


Fuente. Elaboración propia.

6. Para i de 1 a n-1, situar el origen del sistema $\{Si\}$ (solidario al eslabón i) en la intersección del eje Zi con la línea normal común a $Zi-1$ y Zi . Si ambos ejes se cortasen se situaría {Si } en el punto de corte. Si fuesen paralelos $\{Si\}$ se situaría en la articulación i+
7. Situar Xi en la línea normal común a $Zi-1$ y Zi
8. Situar Yi de modo que forme un sistema dextrógiro con Xi y Zi. Figura 40.

Figura 40

Paso 6, 7 y 8



Fuente. Elaboración propia.

Obtener los ángulos y desplazamiento necesarios de cada articulación para la construcción de la tabla y las matrices de acuerdo a los siguientes datos:

θi es el Angulo entre $Xi-1$ y Xi sobre el eje $Zi-1$

dies la distancia medida $Xi-1$ y Xi sobre el eje $Zi-1$

ai es la distancia medida $Zi-1$ y Zi sobre el eje Xi

αi es el Angulo entre $Zi-1$ y Zi sobre el eje $Xi-1$

En la tabla 4 podemos observar el resultado de organizar los datos de cada articulación de acuerdo con el algoritmo de Denavit Hartenberg

Tabla 4

Tabla de Parámetros Algoritmo Denavit Hartenberg

Articulación	θ_i	d_i	a_i	α_i
1	q1	E1	Ex	90
2	90+q2	0	E2	0
3	-90+q3	0	E3	0
4	q4	0	E4	0

*Nota. L*a tabla anterior ayuda a la obtención de los parámetros de rotación para cada una de las articulaciones. *Fuente.* Elaboración propia.

Formar las matrices Homogéneas.

Como último paso hallamos la matriz de transformación homogénea para cada articulación basándonos en la siguiente formula.

$$Ai= Rotz(\theta_i)T(0,0,d)T(a_i,0,0)Rotx(\alpha i)$$

En donde $Rotz(\theta_i)$ es la matriz de rotación en el eje Z con respecto a θ

$T(0,0,d)$ es la matriz de desplazamiento con respecto al eje Z

$T(a_i,0,0)$ es la matriz de desplazamiento con respecto al eje X

$Rotx(\alpha i)$ es la matriz de rotación con respecto al eje X

$$Ai= [C\theta i - S\theta i00S\theta iC\theta i0000100001] * [10000100001di0001] * [100ai010000100001]$$

$$* [10000C\alpha i - S\alpha i00S\alpha iC\alpha i00001]$$

Al realizar la multiplicación de las matrices tenemos como resultado

$$Ai= [C\theta i - C\alpha iS\theta iS\alpha iS\theta iaiC\theta iS\theta iC\alpha iC\theta i - S\alpha iC\theta iaiS\theta i0S\alpha iC\alpha idi0001]$$

Procedemos a encontrar la matriz homogénea para cada articulación con respecto a la tabla.

Articulación Q1

En la tabla 5 se puede observar la tabla para la obtención de la matriz homogénea de la articulación Q1

Tabla 5

Parámetros Articulación Q1

Articulación	$\boldsymbol{\theta i}$	$\boldsymbol{di}$	$\boldsymbol{ai}$	$\boldsymbol{ai}$
1	q1	E1	Ex	90

*Nota. P*arámetros articulación Q1

$$Ai= [C\theta i - C\alpha iS\theta iS\alpha iS\theta iaiC\theta iS\theta iC\alpha iC\theta i - S\alpha iC\theta iaiS\theta i0S\alpha iC\alpha idi0001]$$

$$0A1= [\cos(q1) - \cos(90)\sin(q1)\sin(90)\sin(q1)Ex\cos(q1)\sin(q1)\cos(90)\cos(q1) - \sin(90)\cos(q1)Ex\sin(q1)0\sin(90)\cos(90)E10001]$$

Resolviendo la matriz anterior tenemos que

$$0A1= [\cos(q1)0\sin(q1)Ex\cos(q1)\sin(q1)0 - \cos(q1)Ex\sin(q1)010E10001]$$

Articulación Q2

En la tabla 6 se puede observar la tabla para la obtención de la matriz homogénea de la articulación Q2

Tabla 6

Parámetros Articulación Q2

Articulación	$\boldsymbol{\theta i}$	$\boldsymbol{di}$	$\boldsymbol{ai}$	$\boldsymbol{\alpha i}$
2	90+q2	0	E2	0

Nota. Según el los pasos del algoritmo el Angulo de rotación es 90+q2

$$Ai= [C\theta i - C\alpha iS\theta iS\alpha iS\theta iaiC\theta iS\theta iC\alpha iC\theta i - S\alpha iC\theta iaiS\theta i0S\alpha iC\alpha idi0001]$$

$$1_{A_2} = \begin{bmatrix} \cos(90+q2) & -\cos(0)\sin(90+q2) & \sin(0)\sin(90+q2) & E2\cos(90+q2) \\ \sin(90+q2) & \cos(0)\cos(90+q2) & -\sin(0)\cos(90+q2) & E2\sin(90+q2) \\ 0 & \sin(0) & \cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Resolviendo la matriz anterior tenemos que:

$$1A2= \begin{bmatrix} \cos(90+q2) & -\sin(90+q2) & 0 & E2\cos(90+q2) \\ \sin(90+q2) & \cos(90+q2) & 0 & E2\sin(90+q2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

En donde tenemos como identidades trigonométricas que:

$$\cos(a+b) = \cos(a)\cos(b) - sen(a)sen(b)$$

$$\sin(a+b) = sen(a)\cos(b) + sen(b)\cos(a)$$

$$\cos(90+q2) = \cos(90)\cos(q2) - sen(90)sen(q2) = -sen(q2)$$

$$sen(90+q2) = sen(90)\cos(q2) + sen(q2)\cos(90) = \cos(q2)$$

$$1A2= \begin{bmatrix} -sen(q2) & -\cos(q2) & 0 & -E2sen(q2) \\ \cos(q2) & -sen(q2) & 0 & \text{E2}\cos(q2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Articulación Q3

En la tabla 7 se puede observar la tabla para la obtención de la matriz homogénea de la articulación Q3

Tabla 7

Parámetros Q3

Articulación	$\boldsymbol{\theta i}$	$\boldsymbol{di}$	$\boldsymbol{ai}$	$\boldsymbol{ai}$
3	-90+q3	0	E3	0

Nota. Según él los pasos del algoritmo el Angulo de rotación es -90+q2

$$Ai = \begin{bmatrix} C\theta_i & -C\alpha_i S\theta_i & S\alpha_i S\theta_i & a_i C\theta_i \\ S\theta_i & C\alpha_i C\theta_i & -S\alpha_i C\theta_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$2A3$

$$\begin{bmatrix} \cos(-90+q3) & -\cos(0)\sin(-90+q3) & \sin(0)\sin(-90+q3) & E3\cos(-90+q3) \\ \sin(-90+q3) & \cos(0)\cos(-90+q3) & -\sin(0)\cos(-90+q3) & E3\sin(-90+q3) \\ 0 & \sin(0) & \cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Resolviendo la matriz tenemos que

$$2A3 = \begin{bmatrix} \cos(-90+q3) & -\sin(-90+q3) & 0 & E3\cos(-90+q3) \\ \sin(-90+q3) & \cos(-90+q3) & 0 & E3\sin(-90+q3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\cos(a+b) = \cos(a)\cos(b) - sen(a)sen(b)$$

$$\sin(a+b) = sen(a)\cos(b) + sen(b)\cos(a)$$

$$\cos(-90 + q3) = \cos(-90)\cos(q3) - sin(-90)sin(q3) = \sin(q3)$$

$$\sin(-90 + q3) = sen(-90)\cos(q3) + sen(q3)\cos(-90) = -1$$

$$2A3 = \begin{bmatrix} \sin(q3) & 1 & 0 & E3\sin(q3) \\ -1 & \sin(q3) & 0 & -E3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Articulación Q4

En la tabla 8 se puede observar la tabla para la obtención de la matriz homogénea de la articulación Q4

Tabla 8

Parámetros Q4

Articulación	$\boldsymbol{\theta i}$	$\boldsymbol{di}$	$\boldsymbol{ai}$	$\boldsymbol{ai}$
4	q4	0	E4	0

Nota. Según él los pasos del algoritmo el Angulo serie el valor que se le asigne a q4

$$Ai = \begin{bmatrix} C\theta_i & -C\alpha_i S\theta_i & S\alpha_i S\theta_i & a_i C\theta_i \\ S\theta_i & C\alpha_i C\theta_i & -S\alpha_i C\theta_i & a_i S\theta_i \\ 0 & S\alpha_i & C\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$3A4 = \begin{bmatrix} \cos(q4) & -\cos(0)\sin(q4) & \sin(0)\sin(q4) & E4\cos(q4) \\ \sin(q4) & \cos(0)\cos(q4) & -\sin(0)\cos(q4) & E4\sin(q4) \\ 0 & \sin(0) & \cos(0) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Resolviendo tenemos que:

$$3A4= \begin{bmatrix} \cos(q4) & -\sin(q4) & 0 & E4\cos(q4) \\ \sin(q4) & \cos(q4) & 0 & E4\sin(q4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Por último, tenemos que T el cual es la variable con la ubicación del efector final será igual a

$$T = 0A1 * \ 1A2 * \ 2A3 * \ 3A4$$

$$T = \begin{bmatrix} \cos(q1) & 0 & \sin(90) & Ex\cos(q1) \\ \sin(q1) & 0 & -\cos(q1) & Ex\ \sin(q1) \\ 0 & 1 & 0 & E1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$* \begin{bmatrix} \cos(90+q2) & -\sin(90+q2) & 0 & E2\cos(90+q2) \\ \sin(90+q2) & \cos(90+q2) & 0 & E2\sin(90+q2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$* \begin{bmatrix} \sin(q3) & 1 & 0 & E3\sin(q3) \\ -1 & \sin(q3) & 0 & -E3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$* \begin{bmatrix} \cos(q4) & -\sin(q4) & 0 & E4\cos(q4) \\ \sin(q4) & \cos(q4) & 0 & E4\sin(q4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

nx= *cos(q4) * (cos(90+q2) *cos(q1) *sin(q3) + cos(q1)*sin(90+q2)) + sin(q4) * (cos(90+q2) * (cos(q1) – cos(q1)*sin(90+q2)*sin(q3))*

ox$= \cos(q4) * (\cos(90+q2) * \cos(q1) - \cos(q1) * \sin(90+q2 * \sin(q3)) - \sin(q4) * (\cos(90+q2) * \cos(q1) * \sin(q3) + \cos(q1) * \sin(90+q2))$

ax= $sin(90)$

$\boldsymbol{px}$= *E2*cos(90+q2) *cos(q1) + E3*cos(90+q2) *cos(q1)*sin(q3) + E3*cos(q1)*sin(90+q2) + E4*cos(q4)*(cos(90+q2)*cos(q1)*sin(q3) + cos(q1)*sin(90+q2)) + E4*sin(q4)*(cos(90+q2)*cos(q1) – cos(q1)*sin(90+q2)*sin(q3)) + Ex*cos(q1)*

$\boldsymbol{ny} = cos\,(q4) * (cos\,(90 + q2) * sin\,(q1) * sin\,(q3)\ +\ sin\,(90 + q2) * sin\,(q1))\ +$

$sin\,(q4) * (cos\,(90 + q2) * sin\,(q1)\ - sin\,(90 + q2) * sin\,(q1) * sin\,(q3)$

*oy= cos(q4) *(cos(90+q2) *sin(q1) – sin(90+q2) *sin(q1) *sin(q3)) – sin(q4) * (cos(90+q2) *sin(q1)*sin(q3) + sin(90+q2)*sin(q1))*

$ay =$ *-cosq1*

$py = E2 * cos(90 + q2)\, sin(q1) +\ E3 * cos(90 + q2) * sin(q1) * sin(q3) +\ E3 * sin(90 + q2) * sin(q1) + E4 * cos(q4) * (cos(90 + q2) * sin(q1) * sin(q3) + sin(90 + q2) * sin(q1)) + E4 * sin(q4) * (cos(90 + q2) * sin(q1) - sin(90 + q2) * sin(q1) * sin(q3)) + Ex * sin(q1)$

*nz= cos(q4) *(-cos(90+q2) + sin(90+q2) *sin(q3)) + sin(q4) *(cos(90+q2) *sin(q3) + sin(90+q2))*

*oz= cos(q4) *(cos(90+q2) *sin(q3) + sin(90+q2)) – sin(q4) *(-cos(90+q2) + sin(90+q2) * sin(q3))*

$az = 0$

$pz = E1 +\ E2 * si\,n(90 + q2) -\ E3 * co\,s(90 + q2) +\ E3 * si\,n(90 + q2) * si\,n(q3) +\ E4 * cos\,(q4) * (-cos\,(90 + q2) +\ sin\,(90 + q2) * sin\,(q3))\ +\ E4 * sin\,(q4) * (cos\,(90 + q2) * sin\,(q3)\ + sin\,(90 + q2)$

Para comprobar el resultado del cálculo de las matrices se ha utilizado el lenguaje de programación Python en conjunto de la librería sympy. (Apéndice)

Programa de Escritorio y Renderizado de Gráficos

En esta sesión explicaremos los pasos que se han llevado a cabo para la elaboración del programa de escritorio en donde el usuario tendrá la posibilidad de visualizar en tiempo real los movimientos de cada una de las articulaciones del robot. El programa contara con una serie de elementos informativos y un entorno virtual para dar la sensación de estar ejecutando las rotaciones del robot en un entorno real. A continuación, daremos a conocer como se ha realizado mediante el lenguaje de programación Python, la biblioteca para el diseño de interfaces QtDesigner y la Api de gráficos 3D OpenGL.

Python es un lenguaje multiparadigma, pero también multiplataforma, esto quiere decir que podemos realizar programas que corran en diferentes sistemas operativos como pueden ser: Windows, Linux, Android y Web.

Para la elaboración de nuestro programa hemos de utilizar la biblioteca grafica PYQT5 que nos permite realizar previamente un diseño o boceto gráfico de un programa mediante Qtdesigner. En la figura 41 podemos observar el logo del software Qtdesigner.

Figura 41

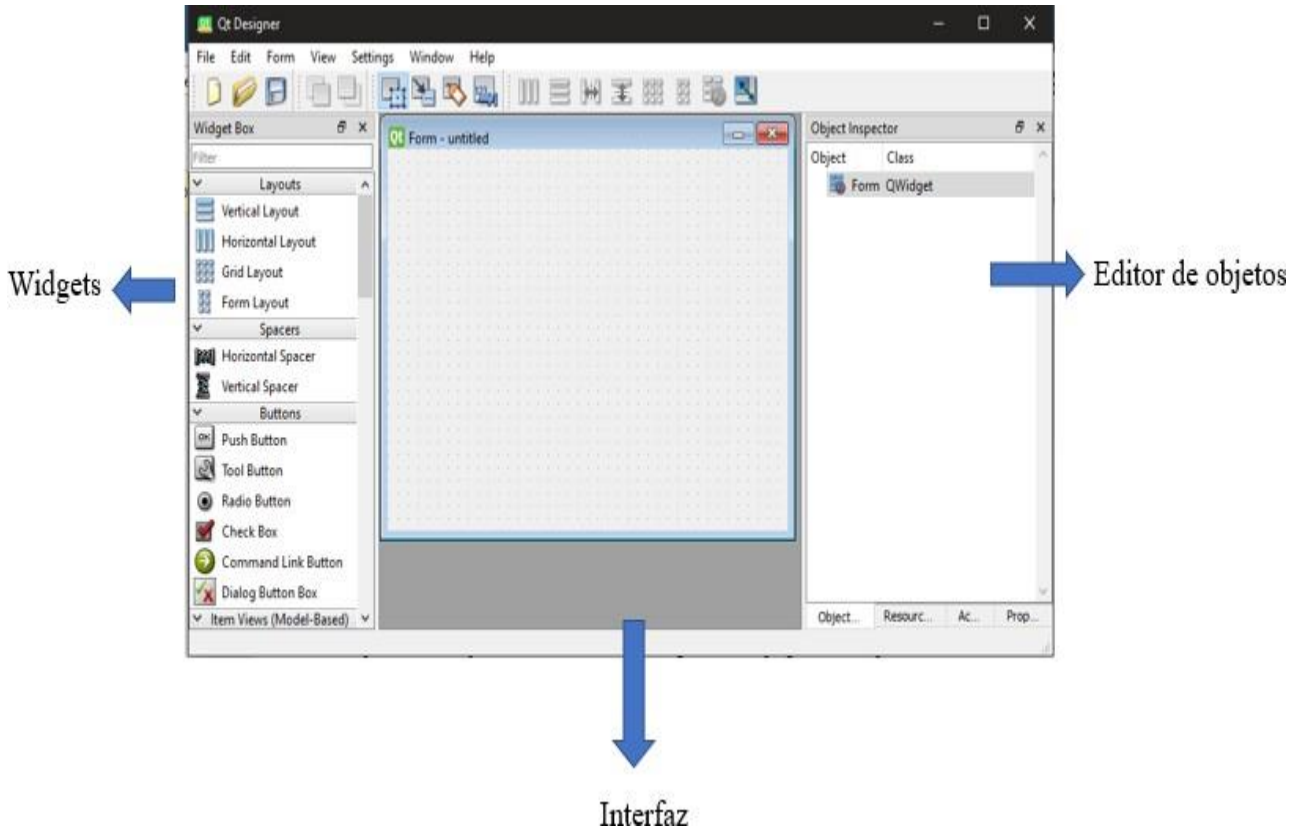
Logo Qtdesigner

Fuente. https://forum.qt.io/

En la figura 42 se presenta la pantalla de inicio en QtDesigner. En la barra de opciones izquierda podemos escoger los widgets como botones, gráficos, labels, tablas, menús y organizadores, los cuales se pondrán en la interfaz del programa.

Figura 42

Pantalla de Inicio en Qtdesigner



Fuente. Elaboración propia.

En la mitad de la ventana podemos observar la interfaz o lienzo en donde se organizarán todos los widgets e imágenes para colocar en nuestra interfaz.

Widgets en Qt Designer

Como mencionábamos anteriormente, este software de distribución gratuita contiene una serie de widgets (Qtdesigner, 2023). Los widgets contienen funciones para ser puestas en nuestro programa. En la tabla 9 se presentan los widgets más importantes:

Tabla 9

Tabla de Descripción Widgets Qtdesigner

Objeto	Definición
PushButton	Los botones nos permiten ingresar a eventos o realizar acciones al ser presionados en nuestro programa
LineEdit	Las líneas de texto modificables nos permiten ingresar valores numéricos o caracteres para luego ser utilizados en nuestro programa
Horizontal Slider	Las sliders bar nos permiten realizar incrementos o decrementos de variables al ser deslizadas

Objeto	Definición
Spin Box	Los spin box nos permiten escoger entre varias opciones ya sean números o caracteres.
Lcd Number	Los widgets LCDNUMBER nos permite visualizar una variable en pantalla al igual que un display 7 segmentos físico.
Label	Los label o etiquetas nos permiten poner texto o imágenes en la interaz
OpenGL Widget	El widget de OpenGL es nuestro widget más importante. Es aquí en donde se pondrá todos nuestros datos de visualización del brazo robot de forma gráfica, OpenGL nos permite realizar aplicaciones 2D y 3D

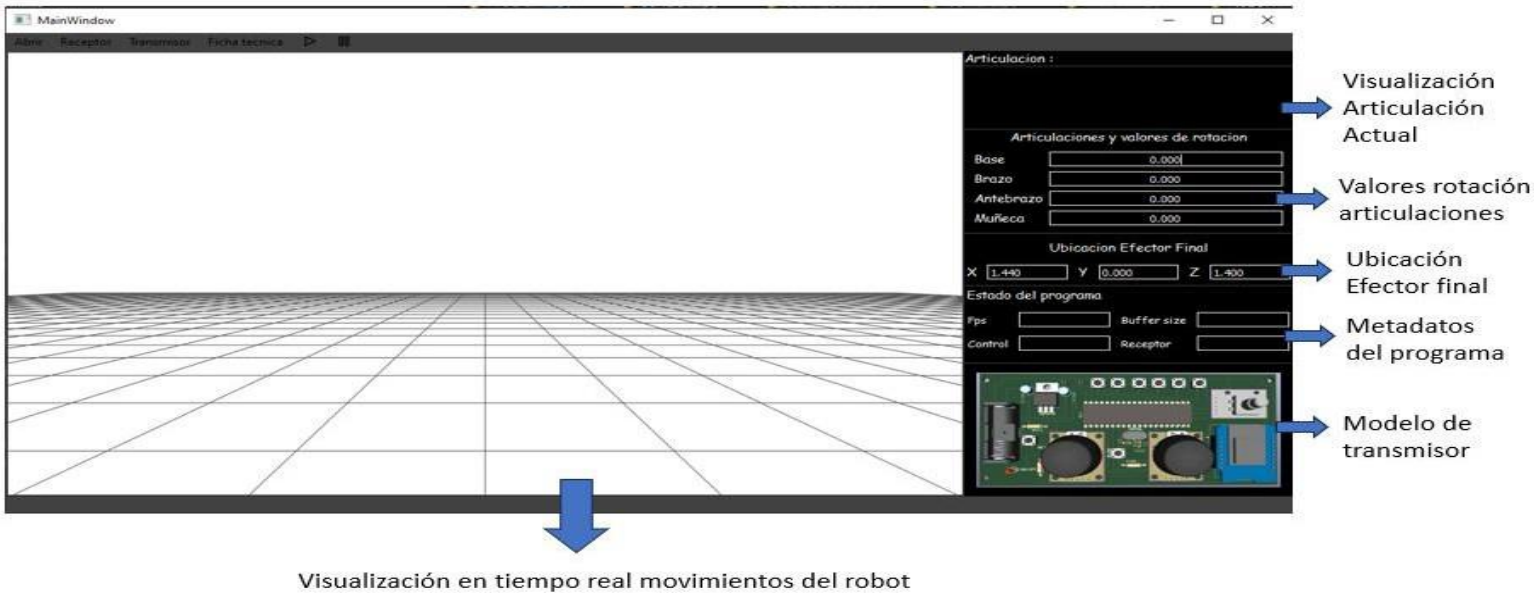
Nota. Los widgets mencionados anteriormente son solo alguno de las múltiples opciones para la creación de interfaces graficas por Qtdesigner. *Fuente*. Autor del proyecto

Diseño de Interfaz

En la figura 43 podemos observar el diseño final de la interfaz en donde se simularán los movimientos de nuestro brazo Robot diseñada en el software Qt Designer.

Figura 43

Diseño de Interfaz para la Ejecución de la Simulación de Movimiento del Brazo Robot



Fuente. Elaboración propia

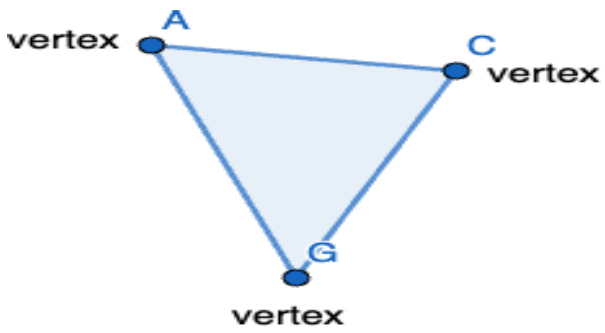
OpenGL

Es una API de gráficos 2D y 3D que permite crear programas de simulación, juegos, motores gráficos y todo lo correspondiente a gráficos por computadora. Su funcionamiento se basa en él envió de datos a la memoria RAM o GPU del computador para así optimizar los procesos llevados a cabo por la CPU. En la actualidad existen varias compañías de diseño de tarjetas y controladores gráficos como lo son, Nvidia, MSI, AMD, Gigabyte etc.

En una escena 3D o 2D todos los elementos que la componen estas compuestos por vértices y polígonos. En la figura 44 podemos observar un triángulo el cual es el polígono compuesto de 3 vértices

Figura 44

Triangulo con 3 Vértices

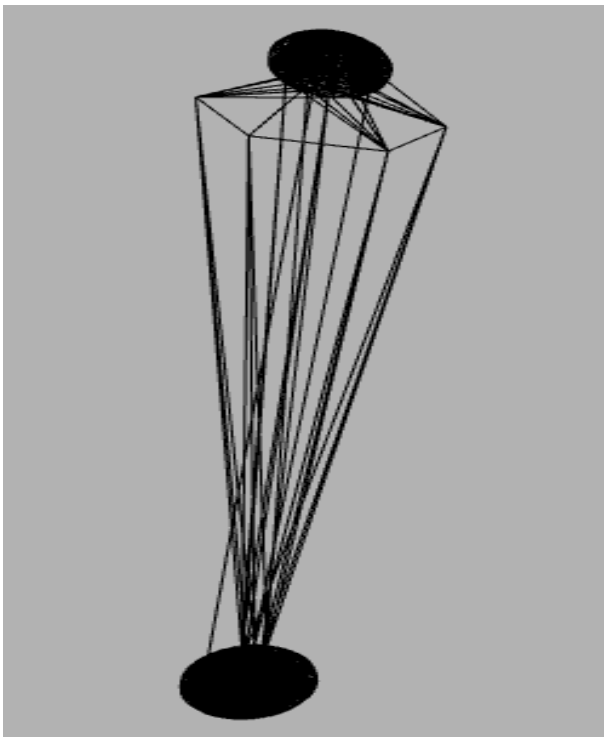


Fuente. https://study.com/learn/lesson/polygons-symmetry-vertices.html

Es posible crear gráficos más complejos uniendo los vértices en forma de triángulos, en la figura 45 podemos observar un gráfico 3D complejo compuesto de un gran número de triángulos.

Figura 45

Figura 3D Compleja Creada a partir de Varios Puntos de Vértices

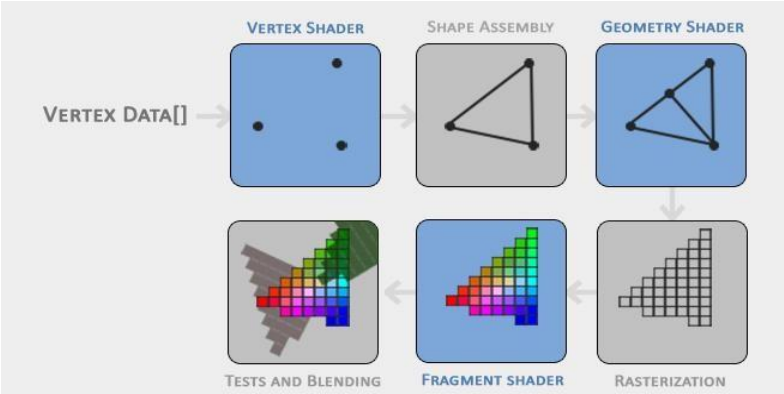


Fuente. Elaboración propia

En OpenGL los datos se manejan de forma 3D. Sin embargo, nuestra pantalla tiene una dimensión en 2D. OpenGL se encarga de transformar las coordenadas 3D de la posición de vértices en pixeles 2D para que luego puedan ser representados en la pantalla. El proceso de transformación de coordenadas 3D a pixeles 2D se llama canalización de gráficos o The Graphics Pipeline (Group L. O., 2022). En la figura 46 podemos observar gráficamente dicho proceso. OpeGL es una API compleja por lo tanto para lograr tener programas gráficos realistas es necesario conocer el proceso que se describirá a continuación

Figura 46

Canalización de Gráficos OpenGL



Fuente. https://learnopengl.com/Getting-started/Hello-Triangle

La entrada Verter data hace referencia los vectores de posición del modelo 3D o 2D. La posición de un vértice en el espacio está dada en OpenGL como un vector de 4 elementos:

$$vec4 = [x, y, z, w]$$

En el ejemplo anterior, figura 44 se formado un triángulo, por lo tanto, es necesario dar a OpenGL un total de 3 vértices.

$$vec1 = [x1, y1, z1, w]$$

$$vec2 = [x2, y2, z2, w]$$

$$vec3 = [x3, y3, z3, w]$$

En donde X, Y y Z, representa la posición con respecto al mundo en OpenGL. Cada uno de estos parámetros está representado por un numero flotante en donde cada número tendrá un peso de 4 bytes en la memoria, por lo tanto, para un vector de 4 posiciones su ocupación en memoria GPU será de 16 bytes. En la figura 47 podemos observar el peso en bytes para vértices de 3 posiciones.

Figura 47

Representación en Bytes de 3 Vértices en Memoria

Fuente. https://learnopengl.com/Getting-started/Hello-Triangle

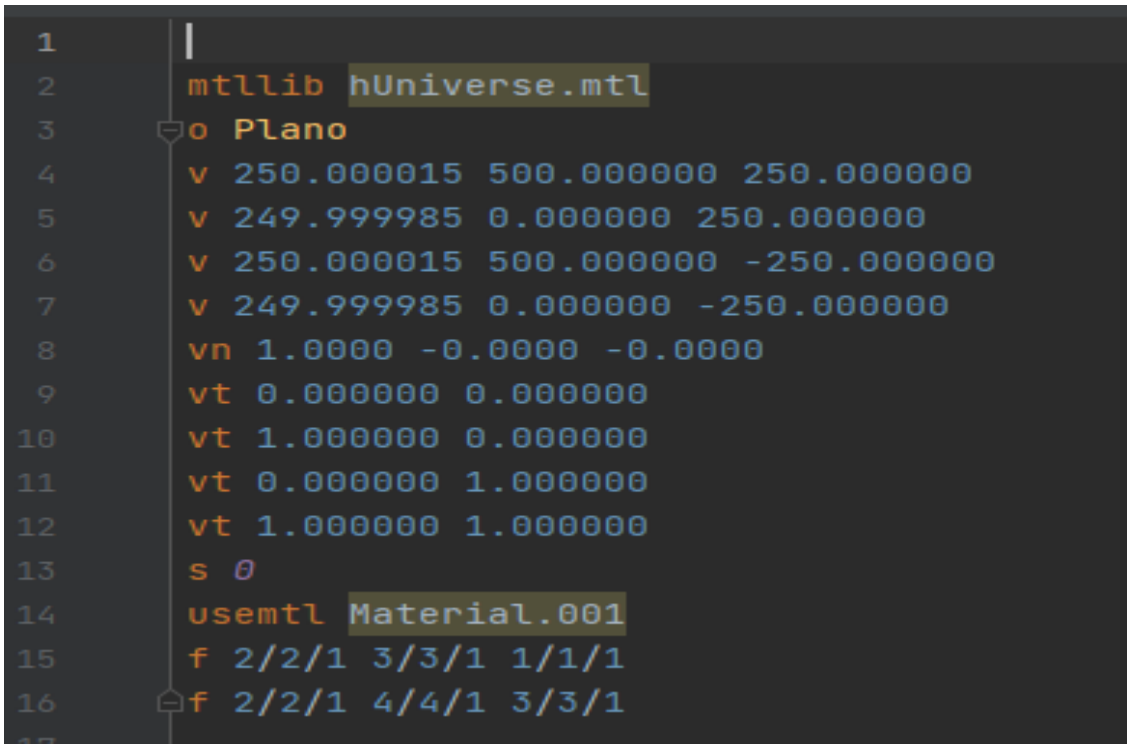
Una vez definidos los 3 puntos de entrada, estos pasan al bloque de Vertex shader. Este bloque aloja los puntos del espacio 3D para luego ser representados como un único atributo de dibujo el cual define el usuario. En este caso como se tiene de ejemplo un triángulo, OpenGL formará esta figura geométrica como se puede observar en la Figura 44.

Por último, en el bloque Fragment shader de la Figura 46 se toma los valores del color de cada vértice, las posiciones de ángulo de la luz que puede ser opcional y se representa el objeto en pantalla. Para el proceso de canalización básico sólo es necesario los bloques Vertex Data, Vertex Shader, Geometry Shader. (Group K. , 2022)

Existen diferentes formatos de archivos 3D, entre ellos uno de los más fáciles y destacados es el .Obj, el cual es un archivo de texto nos da los puntos en el espacio del modelo, “Vértices”, los vectores normales para aplicar luces y sombras y las coordenadas de texturas para poner imágenes en nuestros modelos, en donde las líneas que inician con la letra V son los vértices en las coordenadas X, Y, Z, las líneas con Vn son los vertex normals para aplicar luces y sombras al modelo y las líneas Vt son las coordenadas de textura. Por último, las líneas con F representan el orden en el cual se deben acomodar los datos (Wikipedia, 2021) , Como ejemplo, en la Figura 48 se representa una pared de 250x250cm en un archivo Obj.

Figura 48

Ejemplo de Contenido Archivo Obj

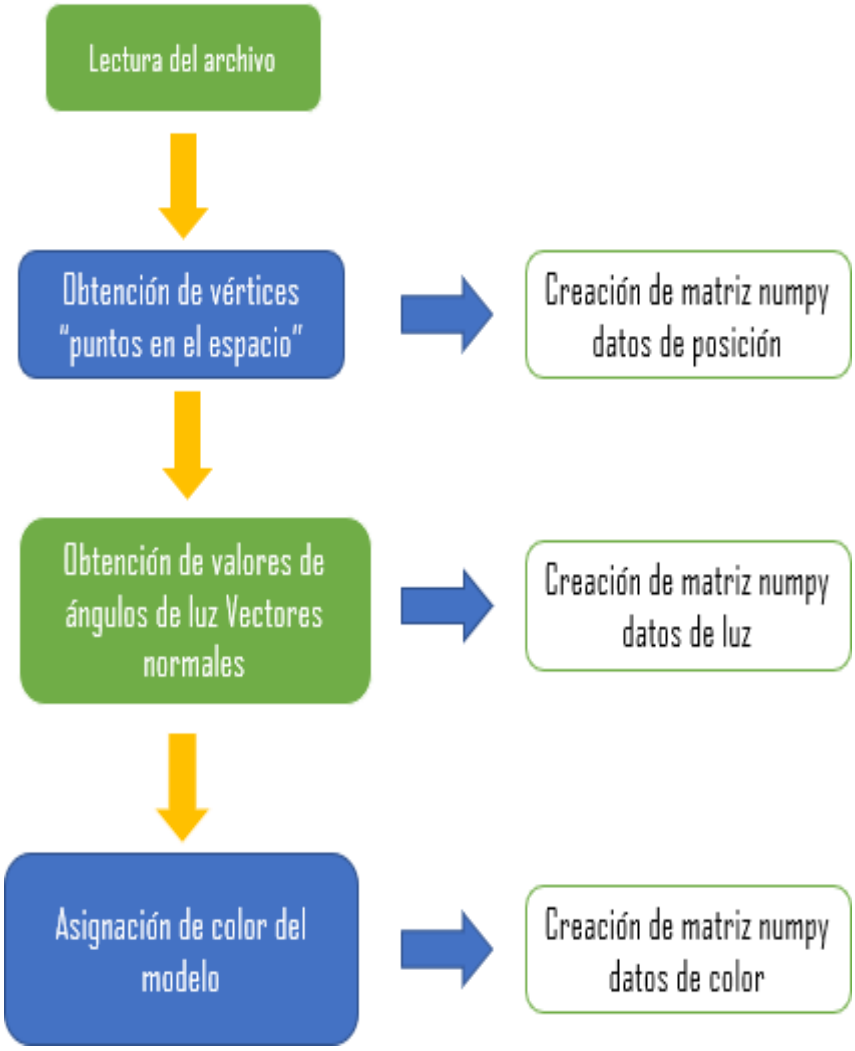


Fuente. Elaboración propia.

Se puede obtener los datos del archivo utilizando la librería para la creación de matrices para Python llamada numpy. Ésta es una biblioteca de código abierto capaz de proporcionar métodos numéricos complejos para el desarrollo de algoritmos y procesos. Actualmente es utilizada junto a Matplolib en alternativa a *MATLAB* por su alto costo. El diagrama de bloques para la obtención de datos en el archivo se presenta en la figura 49.

Figura 49

Algoritmo para Obtener Datos del Modelo

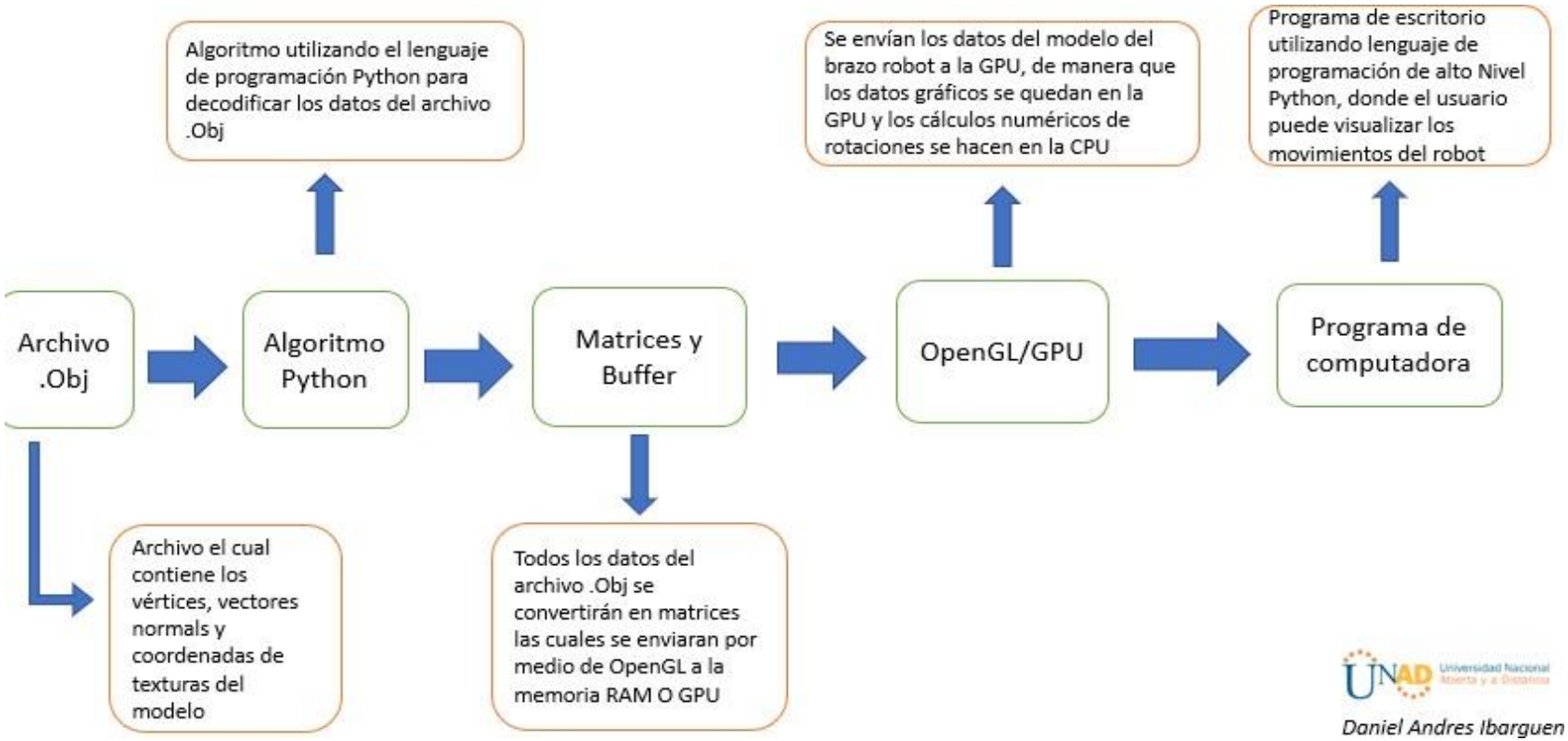


Fuente. Elaboración propia.

Para nuestro modelo de brazo robot utilizado en blender se han exportado el modelo de partes 3D en archivos individuales los cuales se han obtenido los datos de posiciones en el espacio y ángulo de luces. En la Figura 50 se representa el diagrama de bloques para cumplir con este objetivo.

Figura 50

Diagrama de Bloques Metodología Objetivo Numero 2

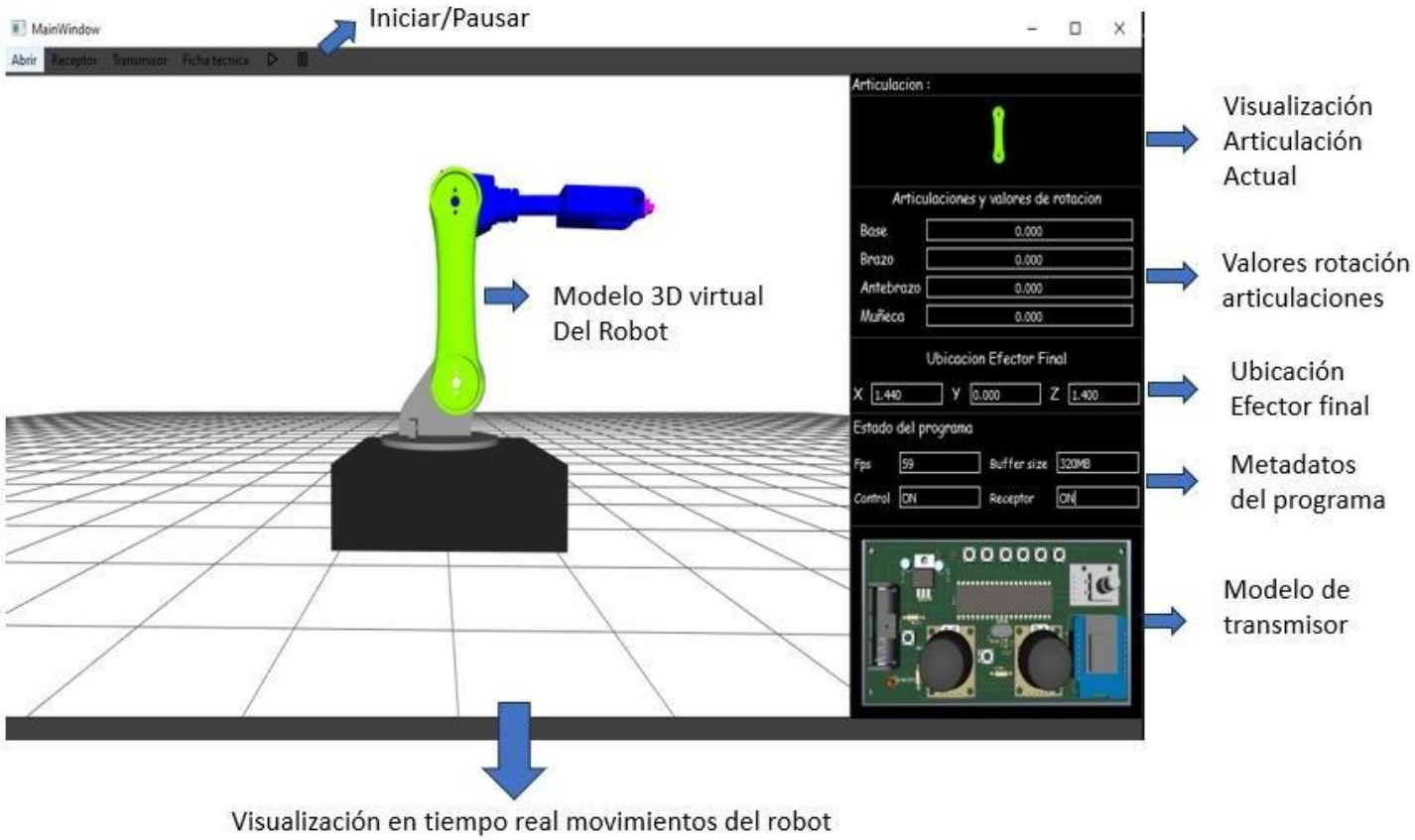


Fuente. Elaboración propia.

En la figura 51 podemos apreciar el resultado final del diseño del software de escritorio para el control de brazo robot

Figura 51

Resultado Metodología 2



Fuente. Elaboración propia.

Diseño del Circuito Transmisor

En esta sesión se explicarán los pasos correspondientes para el diseño del circuito inalámbrico y los componentes electrónicos utilizados para el óptimo funcionamiento del circuito transmisor.

Este circuito le permitirá al usuario realizar los diferentes movimientos de la cámara y las rotaciones de cada una de las articulaciones. La cámara en el software hace referencia a la vista en el espacio que permite observar el modelo del brazo robot manipulador en el entorno virtual (en las próximas secciones de este capítulo se dará a conocer un poco más de este tema). Se han utilizado diferentes periféricos del microcontrolador *PIC16F877A* para poder realizar la captura de datos de los componentes electromecánicos como lo son: joystick, botones y potenciómetro digital. Dentro de los periféricos tenemos Puertos Digitales, Convertidor ADC, Temporizadores, Interfaz de comunicación USART. A continuación, se presenta todo el procedimiento.

PIC16F877A:

Es un microcontrolador de 8 bits que cuenta con todos los periféricos que requiere el proyecto. El sistema de operación es de 20MHZ, en donde la frecuencia de procesamiento interno de la CPU está dada por

$$Frecuencia\ de\ oscilacion = \frac{Fclock}{4} = \frac{20MHz}{4} = 5MHz$$

El tiempo que tardará el microcontrolador para ejecutar cada instrucción dada en el código de programación será:

$$Tiempo\ por\ instruccion = \frac{1}{5MHz} = 0.2\mu S$$

Memoria flash 14.3 Kilobytes en donde se alojará el programa, memoria SRAM de 368 Bytes suficiente para guardar variables enteras de 16 bits o 2 bytes, 33 entradas y salidas

digitales, 8 canales ADC, 2 Temporizadores de 8 bits y uno de 16bits. En la figura 52 se puede apreciar el encapsulado de 40 pines tipo DIP

Figura 52

PIC16F877A DIP 40

Fuente. https://www.microchip.com/

Puerto ADC PIC16F877A

El periférico ADC del microcontrolador es de 10 bits. Su ecuación de resolución o las muestras mínimas de voltaje que puede captar en sus entradas analógicas será:

$$Resolucion\ ADC = \frac{V_{IN}}{2^N - 1}$$

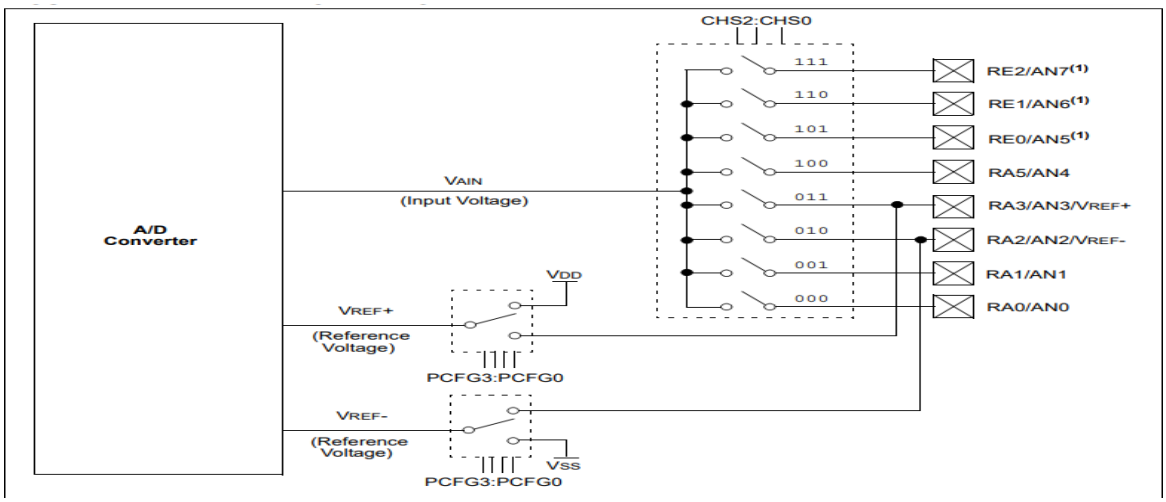
En donde VIN es el voltaje de entrada del canal ADC y su valor no puede superar 5V ni ser menor que 0V. N será igual al número de bits del convertidor ADC del PIC para este caso es de 10. Si trabajamos con un voltaje estándar de 0V a 5V entonces la resolución del ADC será:

$$Resolucion\ ADC = \frac{V_{IN}}{2^N - 1} = 5V210 - 1 = 4.83mV$$

Es decir que nuestro convertidor será capaz detectar cambios en su entrada mínimos de 4.83mV. En la figura 53 se muestra el diagrama de bloques de conversión del convertidor ADC.

Figura 53

Diagrama de Bloques Convertidor ADC



Fuente. https://www.microchip.com/en-us/product/pic16f877a

Del diagrama de bloques anterior podemos observar que de RA0 a RE2 son los pines o entradas analógicas disponibles del microcontrolador. CHS2:CHS0 es el multiplexor para escoger que entrada analógica deseamos leer. Vre+ y Vref- son los valores de referencias máximos y mínimos que queremos darle a nuestro conversor, para nuestro caso como hemos explicado en la ecuación anterior de resolución del ADC se utilizará un voltaje máximo de 5V y mínimo de 0V.

Temporizador de 8 Bits Timer0 del PIC16F877A

El Timer0 del microcontrolador es un periférico contador/temporizador de eventos externos e internos de 8 bits el cual nos permite hacer interrupciones en el código del microcontrolador en tiempo real de acuerdo con el tiempo que se haya programado para que el microcontrolador ejecute tareas primarias o secundarias sin tener que parar el programa.

Modo Contador

En modo contador éste contara pulsos externos o señales cuadradas a través de su pin RA4/TOCK1.

Modo Temporizador

Cuenta los pulsos internos del reloj. Para el proyecto se utilizará el modo temporizador de manera que el microcontrolador pueda leer los cambios del potenciómetro digital enconder cada 20uS. La ecuación de desbordamiento del TIMER0 está dada por:

$$T = Tiempo\ por\ instruccion * Prescaler * (256 - TMR0)$$

En donde

$$Tiempo\ por\ instruccion = 1Frecuencia\ de\ oscilacion$$

El prescaler puede tomar valores de programación de 1, 2, 4, 8, 16, 32, 64, 128 y 256 y TMR0 es el valor de 8 bits que cargamos en el registro del TIMER0. A continuación se da un ejemplo para realizar interrupciones en el microcontrolador cada 500 µs utilizando un cristal de 4MHz y un prescaler de 2

$$Frecuencia\ de\ oscilacion = Fclock4 = \ 4MHz4 = 1MHz$$

$$Tiempo\ por\ instruccion = 11MHz = 1\mu S$$

$$500\mu S = 1\mu S * 2 * (256 - TMR0)$$

$$500\mu S2\mu S= (256 - TMR0)$$

$$250 = (256 - TMR0)$$

$$TMR0 = \ 256 - 250 = \ 6$$

Debemos cargar el Registro TMR0 con el valor de 6, con un cristal de 4MHz y un prescaler de 2 para tener temporizaciones cada 500µS (Brejio, 2008)

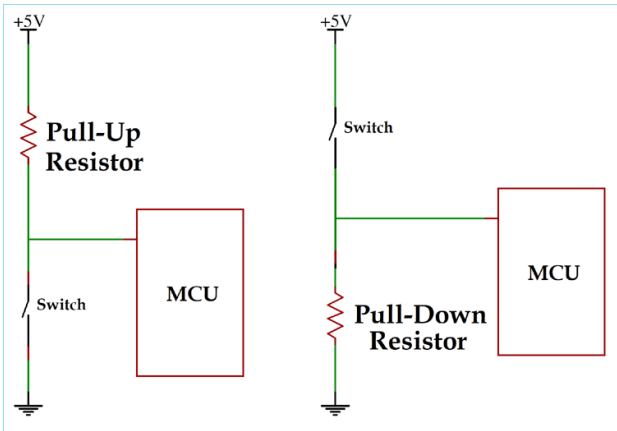
Entradas y Salidas Digitales del PIC16F877A

El microcontrolador contiene un total de 33 pines de entrada y salidas digitales. Si se configuran como entrada es necesario realizar configuraciones de resistencias pull up o pull Down como se muestra en la figura 54, para proteger el PIC de altos pasos de corriente. Los pines de entrada digital se definen por puertos. El PIC16F877A contiene un total de 5 puertos llamados, Puerto A, Puerto B, Puerto C, Puerto D, Puerto E.

Para el proyecto se utiliza el puerto B que contiene 8 entradas y salidas digitales, siendo RB7 el pin con el bit más significativo y RB0 el pin con el bit menos significativo.

Figura 54

Configuración de Resistencias Pull Up y Pull Down



Fuente. https://circuitdigest.com/tutorial/pull-up-and-pull-down-resistor

Puerto Serie Síncrono/Asíncrono USART

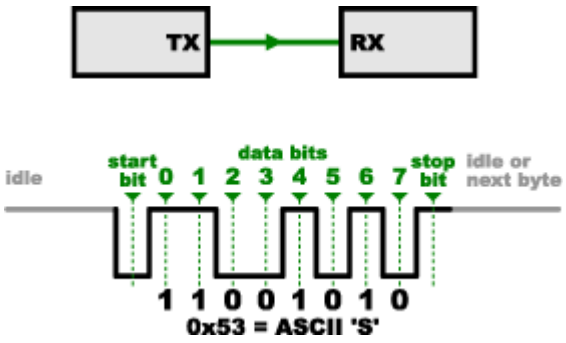
Este periférico nos permite enviar y recibir datos a través de un puerto serie de dos pines como se ha explicado en el marco conceptual y teórico. Este tipo de transmisión permite trabajar a diferentes velocidades que se miden en Baudios. Los baudios son una unidad de medida que se utilizan en las telecomunicaciones para medir el número de veces que el estado de una señal

cambia por segundo. El estado de la señal puede ser todo tipo se alteración que se le pueda realizar a ésta como puede ser: la fase, la frecuencia o el voltaje. (IBM, 2021)

En la figura 55 se muestra el diagrama de bloques de conexión entre dos dispositivos por puerto USART como pueden ser PC – UC o UC – UC en donde “UC” hace referencia a microcontrolador.

Figura 55

Transmisión y Recepción de Datos Serie Puerto USART



Fuente. https://itvoyagers.in/what-is-uart-simple-and-easy-explanation-iot-8/

Joysticks

Para poder mover la cámara que muestra la posición del modelo de brazo robot en el entorno virtual, es necesario mover y rotar la vista, por lo tanto, se hará como si se simulara un video juego. Los josticks Figura 56**,** funcionan con salidas analógicas para los dos ejes que en este caso son **X y Y,** permitiendo obtener estos valores y transformarlos en ángulos Euler para poder rotar la posición de la cámara en nuestro entorno virtual en donde estará nuestro modelo de brazo robot.

Figura 56

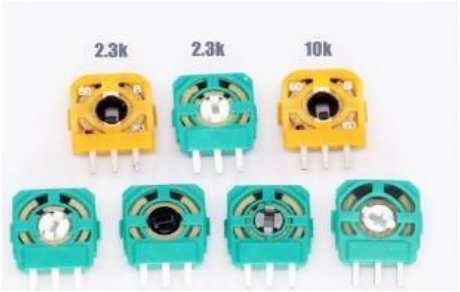
Modelo de Modulo Joystick

Fuente. https://www.vistronica.com/modulos/modulo-joystick-palanca-de-mando-ps2-detail.html

Dentro del joystick se encuentra un sensor de posición resistivo o una resistencia variable que varía su valor dependiendo de la posición de la palanca la cual se encuentra conectada mecánicamente con el sensor de posición. En la figura 57 podemos apreciar los sensores resistivos que se encuentran en el interior de un joystick.

Figura 57

Sensor de Posición Resistivo

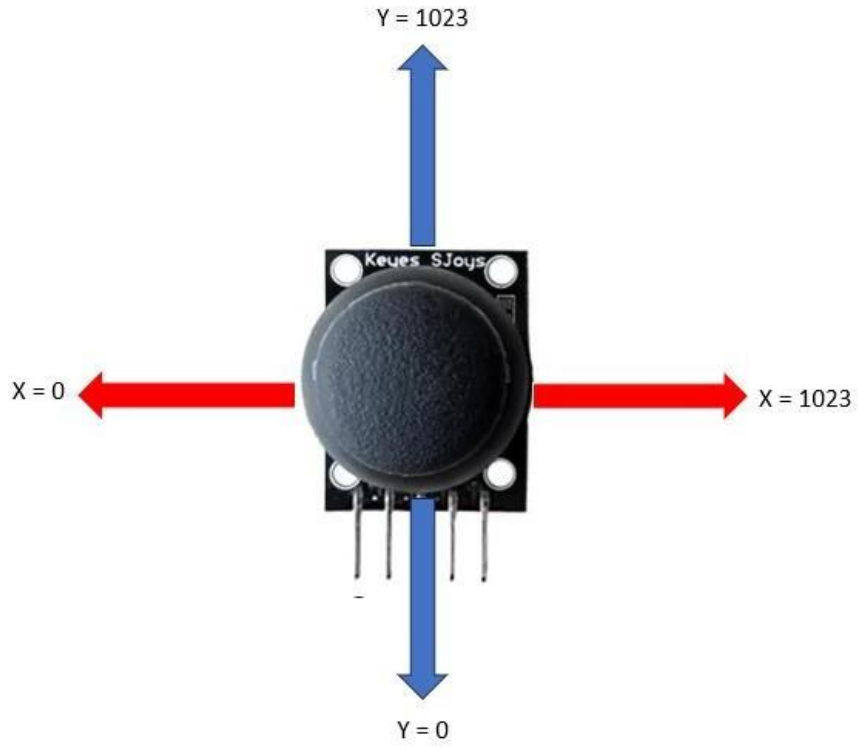


Fuente. https://www.ps84.es/

Los joysticks permiten moverse en los ejes X y Y, por lo tanto, para poder realizar la lectura es necesario leer 2 canales analógicos por cada joystick. En la figura 58 se da a conocer los valores digitales de lectura del convertidor ADC al mover la palanca en cada uno de los extremos del joystick

Figura 58

Valores de los Ejes X y Y del Joystick en los Canales ADC del Microcontrolador

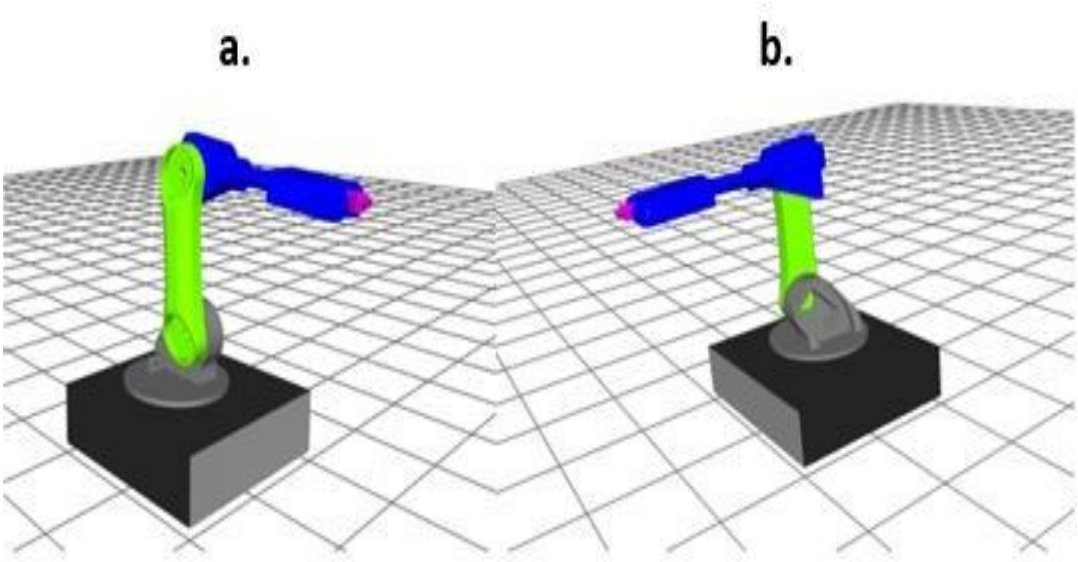


Fuente. Elaboración propia

El joystick ubicado a la izquierda nos servirá para desplazarnos en los ejes X y Y del entorno 3D del brazo robot y el joystick de la derecha nos servirá para rotar la vista de la cámara en los ejes Z y X del entorno. En la figura 59 se muestra diferentes posiciones de cámara posibles al mover los joysticks.

Figura 59

Vista del Brazo Robot desde Diferentes Puntos de la Cámara



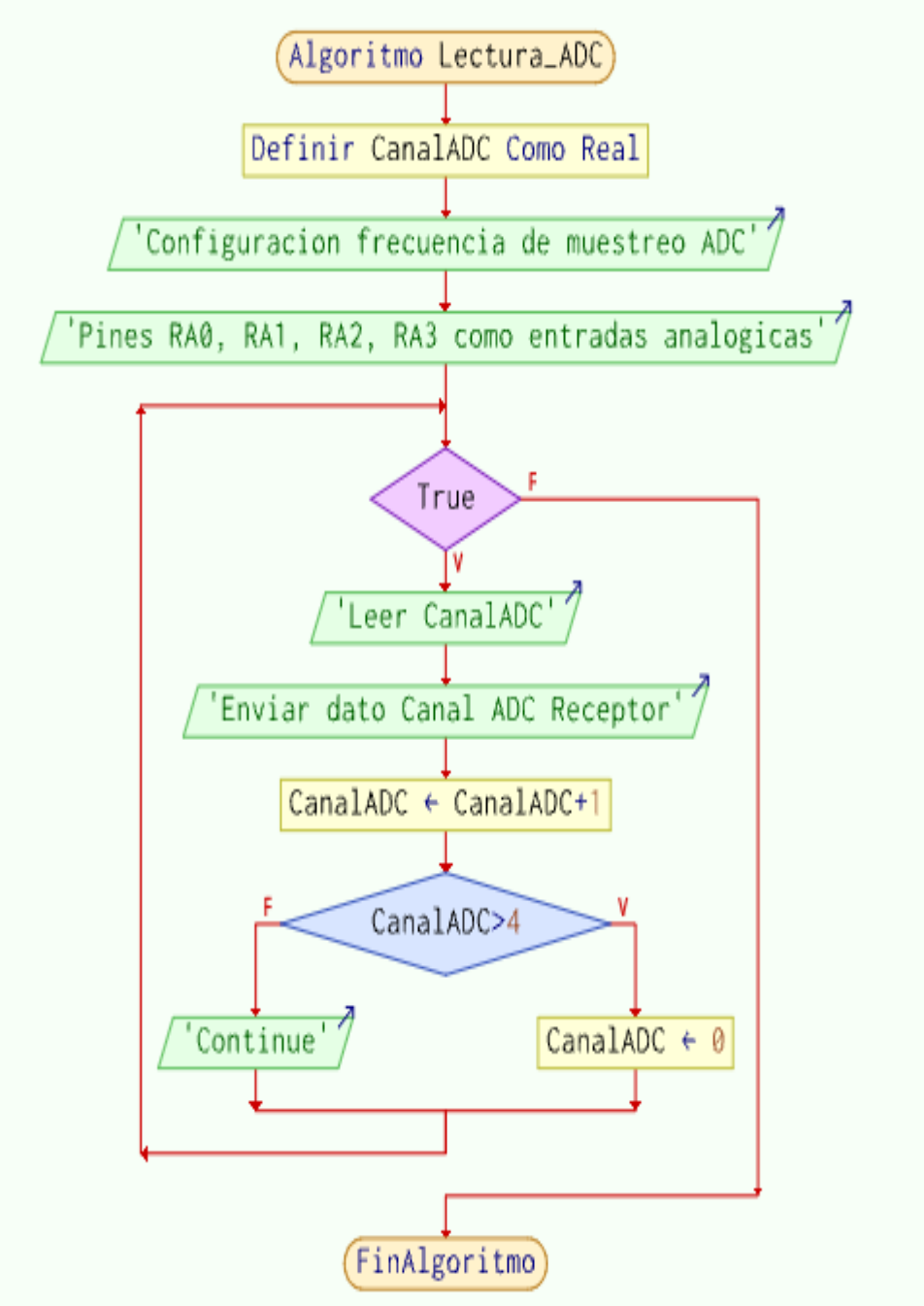
Nota. En figura se puede apreciar cómo se ha cambiado de posición la cámara a. y b.

Fuente. Elaboración propia.

En la figura 60 se muestra el algoritmo para leer los 4 canales analógicos provenientes de los dos joysticks y ser enviados al receptor, teniendo en cuenta que el microcontrolador deberá evaluar los estados de las entradas analógicas se ha dispuesto de un ciclo en donde se obligue al procesador del sistema analizar por medio de su hardware los cambios en los valores analógicos en sus convertidores ADC.

Figura 60

Algoritmo Lectura Canal ADC



Fuente. Elaboración propia

Botones para la Elección de la Articulación del Brazo Robot, Reseteo de Cámara y Articulaciones

Se emplean de diferentes tipos teniendo en cuenta que el funcionamiento de éste puede ser normalmente abierto o cerrado, y su configuración de resistencias puede ser Pull-UP y Pull-Down Figura 61.

Figura 61

Pulsadores de 2 Pines

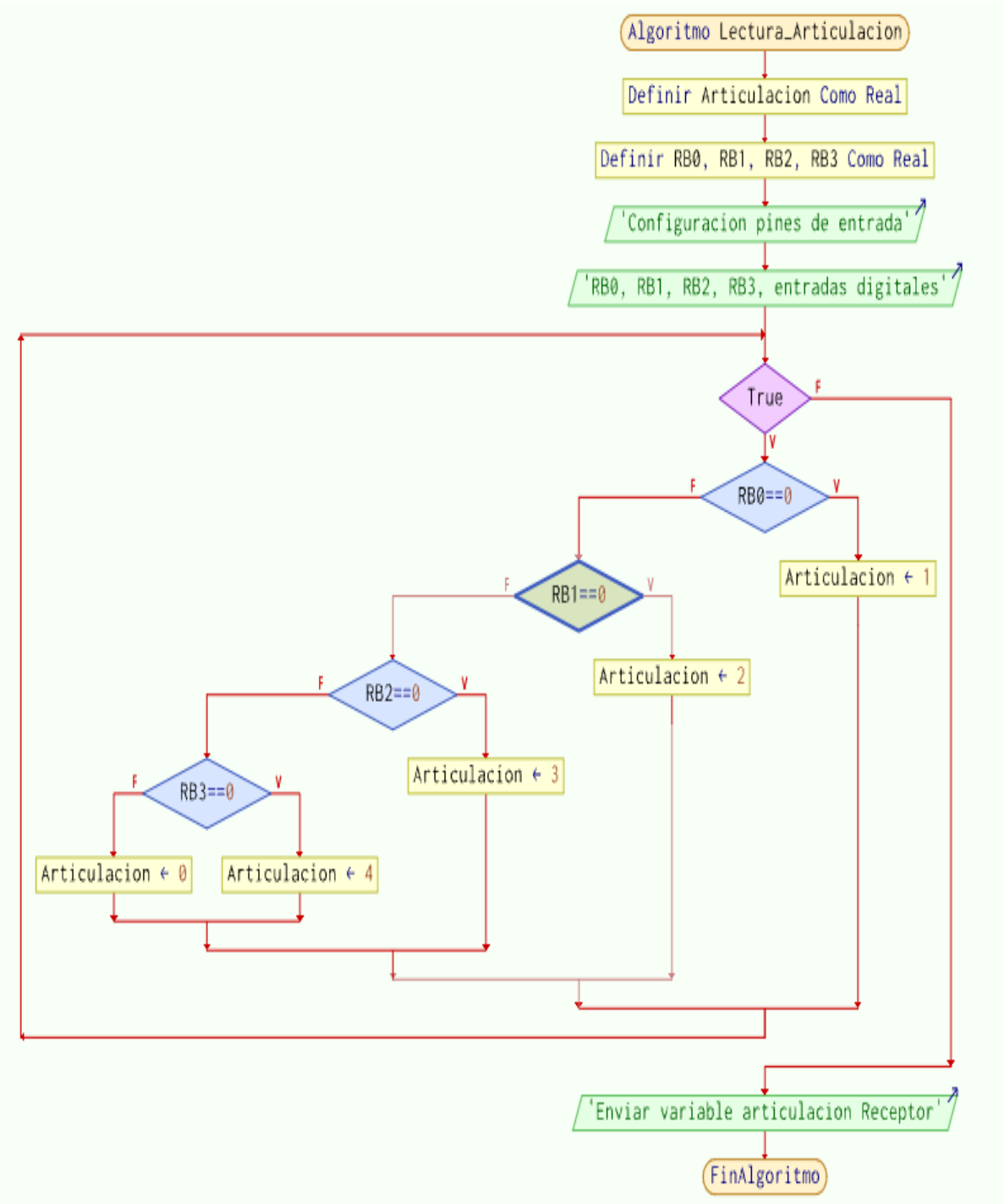
Fuente. https://ssdielect.com/

Al tener 4 articulaciones es necesario tener 4 entradas digitales en el microcontrolador *PIC16F877A.* Al leer estos valores se le asigna un valor numérico de 1 a 4 a una variable y se le envía al receptor para poder rotar la respectiva articulación en el software del computador. En la figura 62 se muestra el diagrama de flujo para la tarea de asignación de articulación a través del microcontrolador.

En caso de que el usuario desee restablecer la posición original de la cámara o establecer la posición de las articulaciones nuevamente en 0, se ha conectado dos botones adicionales también conectados al microcontrolador, con el fin de realizar esta instrucción en el programa. En la figura 63 se presenta el diagrama de flujo para este proceso.

Figura 62

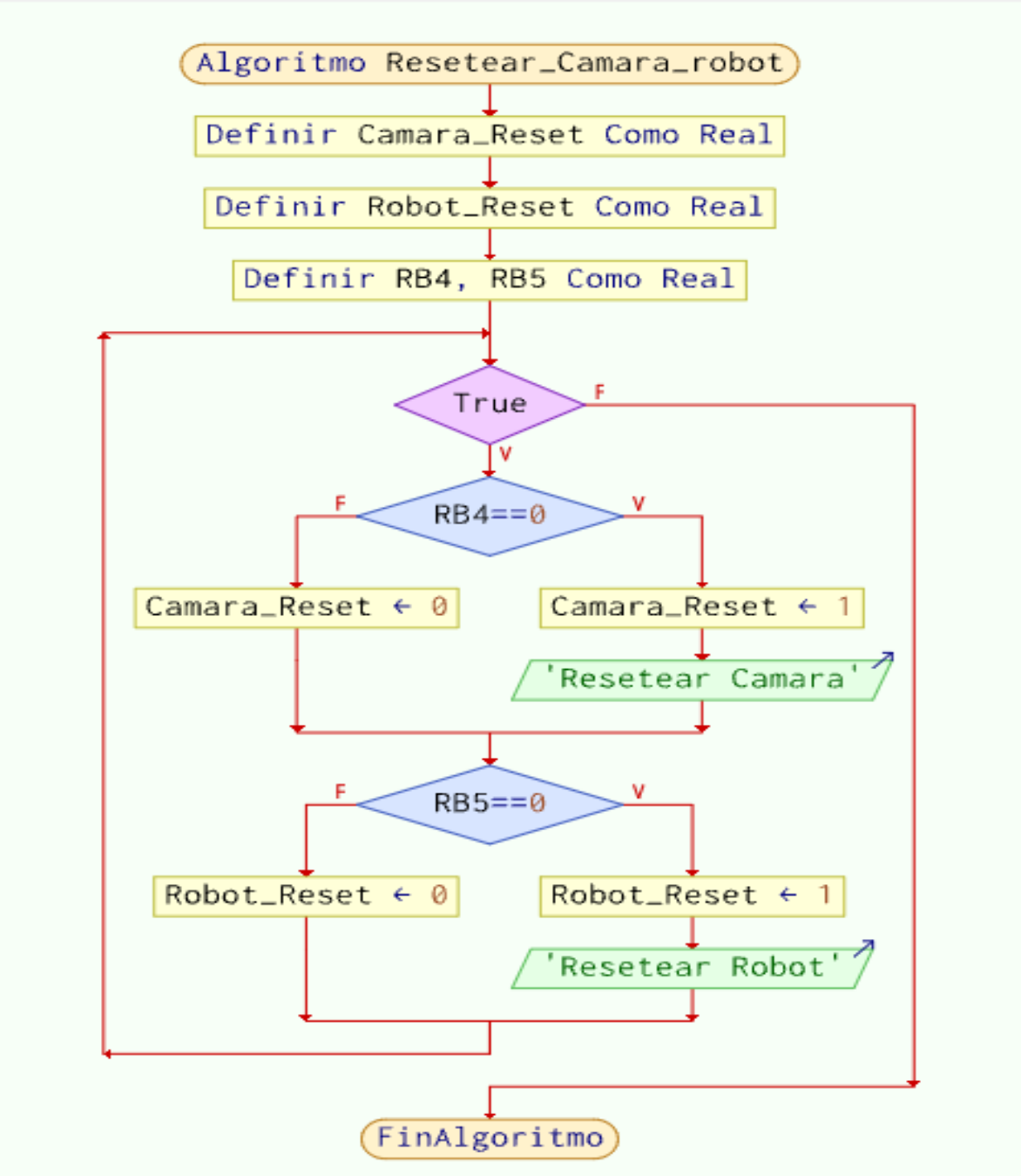
Algoritmo Lectura Botones Articulación



Fuente. Elaboración propia.

Figura 63

Algoritmo para Restablecer Cámara y Articulaciones



Fuente. Elaboración propia.

Potenciómetro Digital

Mediante un potenciómetro giratorio digital de 360 grados se controla el movimiento de las 4 articulaciones del robot. Al girar 360 grados nos da la posibilidad de girar las articulaciones de acuerdo con sus límites y alcances descritos en la tabla 1 “Tabla de especificaciones técnicas”. En la figura 64 se puede ver físicamente el aspecto del potenciómetro digital enconder.

Figura 64

Potenciómetro Rotativo Digital

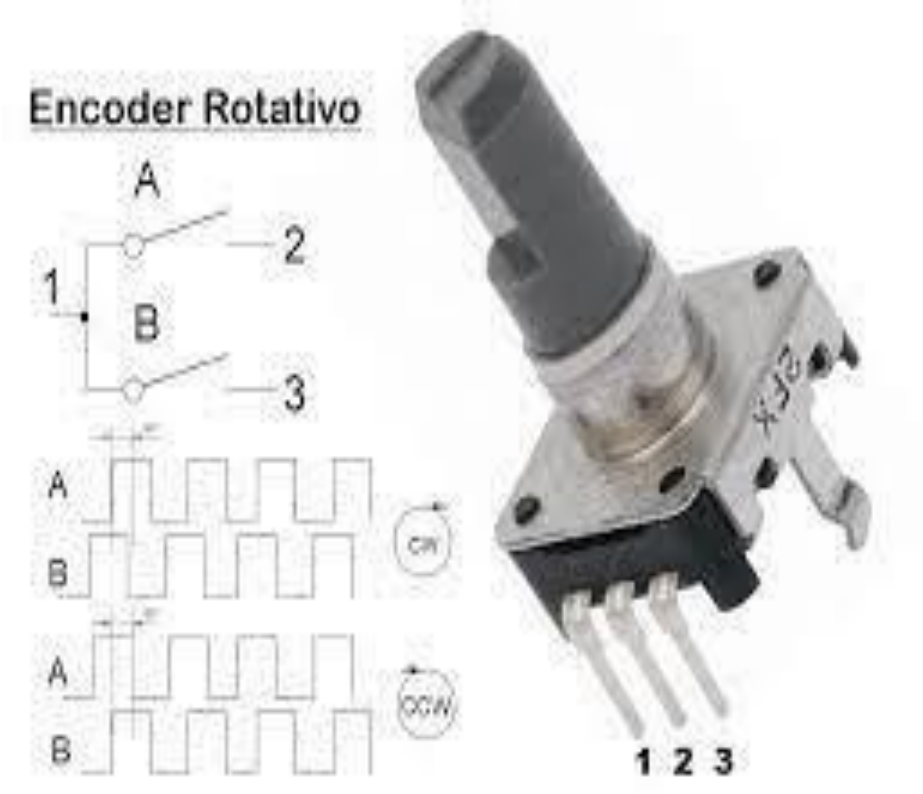
Fuente. https://ssdielect.com/teclados-y-mandos/1426-md-encoder.html

Funcionamiento de un Potenciómetro Digital Rotativo Enconder

Este tipo de componente cuenta con 3 pines en donde el pin 1, es el pin común y el pin 2 y 3 son los contactos que se activan dependiendo de la dirección de rotación del componente. En la figura 65 se puede observar las señales en el tiempo de los pines del potenciómetro encoder. La salida A y B se encuentran desfasadas, es decir al girar el potenciómetro a la derecha la salida B se pondrá en estado alto primero que la A. Al obtener estas dos lecturas en un microcontrolador podemos determinar si el potenciómetro fue girado a la derecha o a la izquierda dependiendo de cual entrada se puso en estado alto primero.

Figura 65

Funcionamiento de un Potenciómetro Encoder



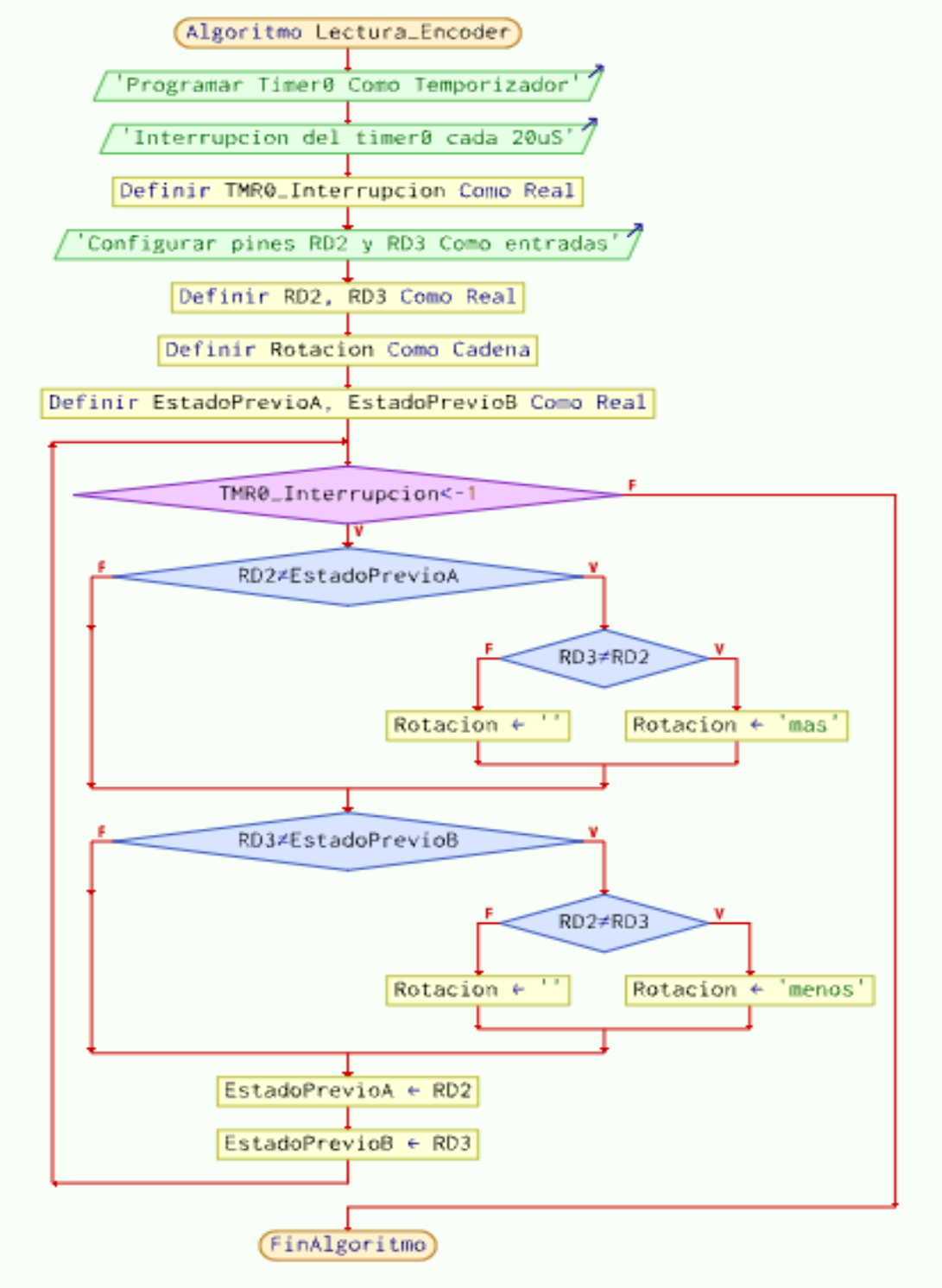
Fuente https://www.makerelectronico.com/producto/potenciometro-encoder-de-rotacion-digital/

Una de las grandes ventajas de este componente frente a un potenciómetro analógico es que podemos girarlo 360 grados y su lectura no depende de valores de voltaje en el tiempo sino de pulsos cuadrados los cuales podemos ir registrando a través de las entradas digitales del microcontrolador.

En la figura 66 se define el algoritmo para la lectura del potenciómetro enconder cada vez que se produce una interrupción en el Timer 0 cada 20 us.

Figura 66

Algoritmo de Lectura Potenciómetro Encoder



Fuente. Elaboración propia.

Chip para la transmisión de datos mediante el uso del ESP8266

Este chip cuenta con un protocolo de comunicación que trabaja bajo la frecuencia de 2,4 GHz llamado ESPNOW. Este protocolo de comunicación es capaz de enviar hasta un total de 250 Bytes por mensaje. Para su funcionamiento es necesario indicarles a los chips de transmisión la dirección MAC del dispositivo receptor. Este chip también puede conectarse a redes de internet vía Wifi, lo cual lo dota de una dirección MAC única. En la figura 67 se presenta el chip de forma física. El protocolo de comunicación acepta el direccionamiento de datos con las siguientes configuraciones:

- Un maestro y un esclavo
- Un maestro y varios esclavos
- Varios maestros y un esclavo

Figura 67

Esp8266 CHIP

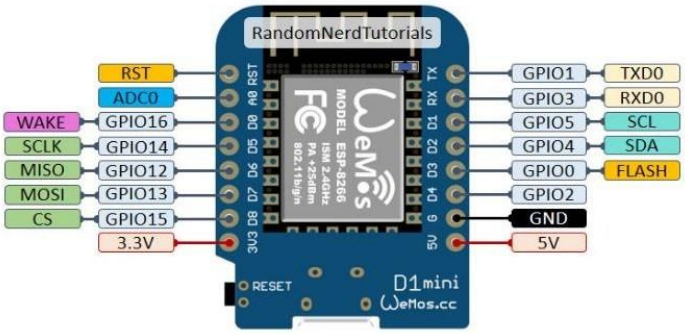
Fuente. https://www.espressif.com/en/products/socs/esp8266

Los datos de los componentes mencionados anteriormente como, los potenciómetros, pulsadores etc., provenientes del PIC16F877A, serán enviados al chip ESP8266 a través de la interfaz USART, de allí el chip transmisor lo enviara finalmente al receptor conectado al computador que contiene el software.

El chip ESP8266 tiene varios módulos para su funcionamiento. Para este proyecto se utilizará el ESP8266 Wemos mini como se puede observar en la figura 68.

Figura 68

ESP8266 Wemos Mini



Fuente. https://tienda.bricogeek.com/wemos-d1-mini/1588-wemos-d1-mini-esp8266-wifi.html

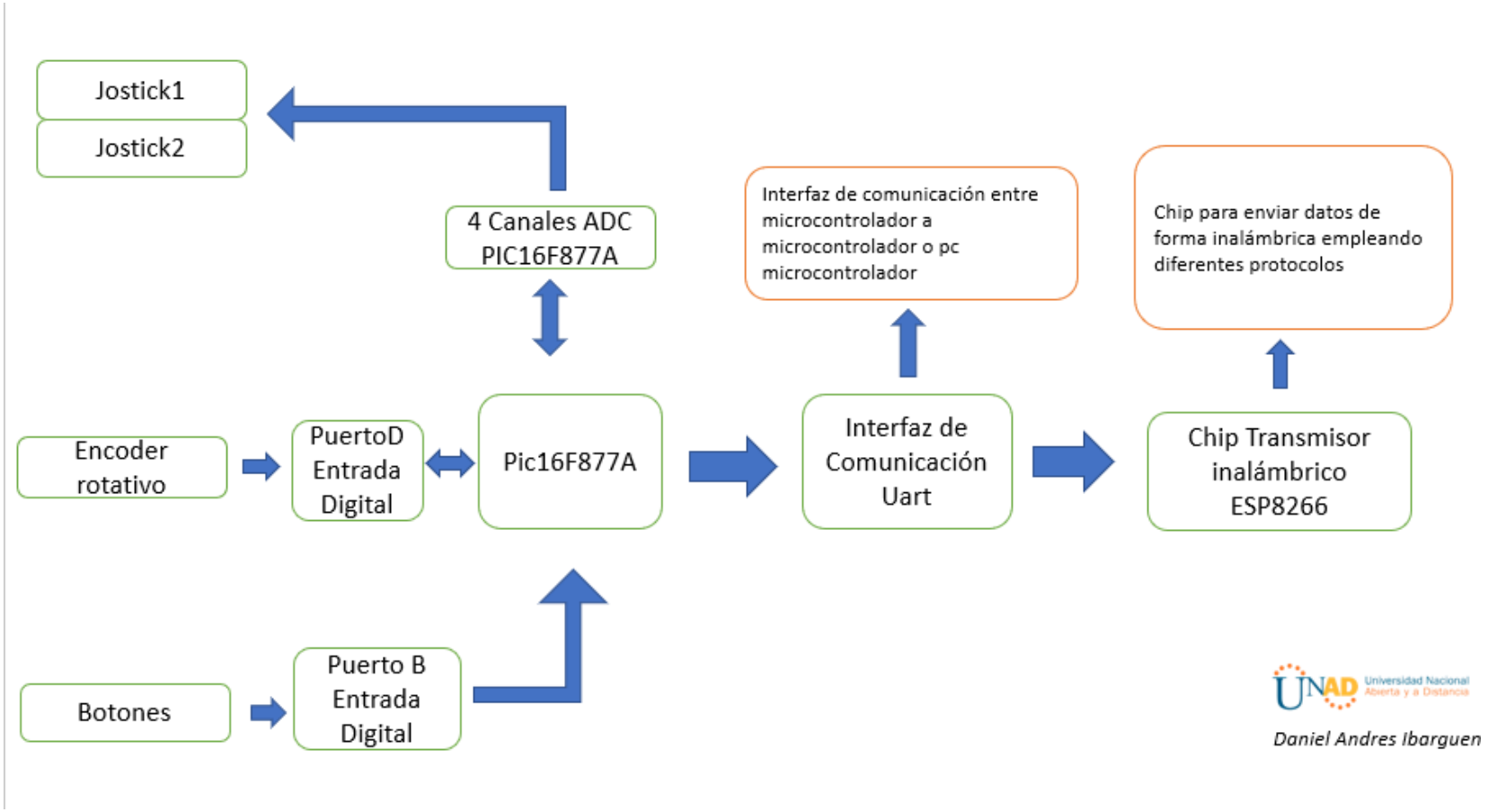
La placa ESP8266 Wemos mini cuenta con las siguientes características:

- Voltaje de alimentación 5V
- Puerto Micro USB B
- Conversor USB-Serial CH340G
- Frecuencia de la CPU 80MHz/160MHz
- Pines de entrada Digitales 11
- Memoria Flash 4MB
- Interfaz USART.
- Conexión Wifi

El diagrama de bloques general para el funcionamiento del circuito transmisor se puede apreciar en la figura 69.

Figura 69

Diagrama de Bloques Metodología Objetivo Específico Numero 3



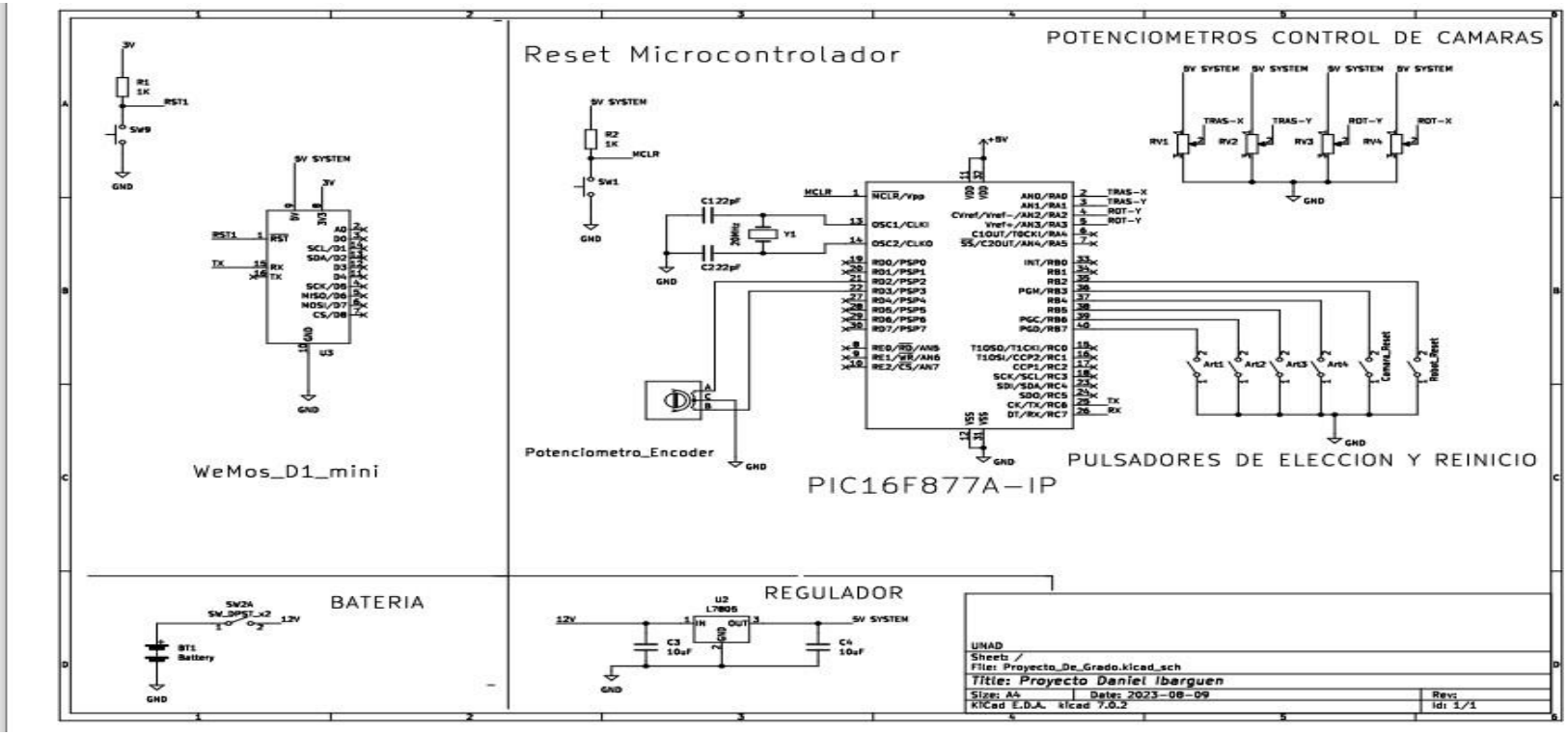
Fuente. Elaboración propia.

Circuito Transmisor

En la figura 70 se puede apreciar el esquema del circuito transmisor

Figura 70

Esquemático Circuito Transmisor



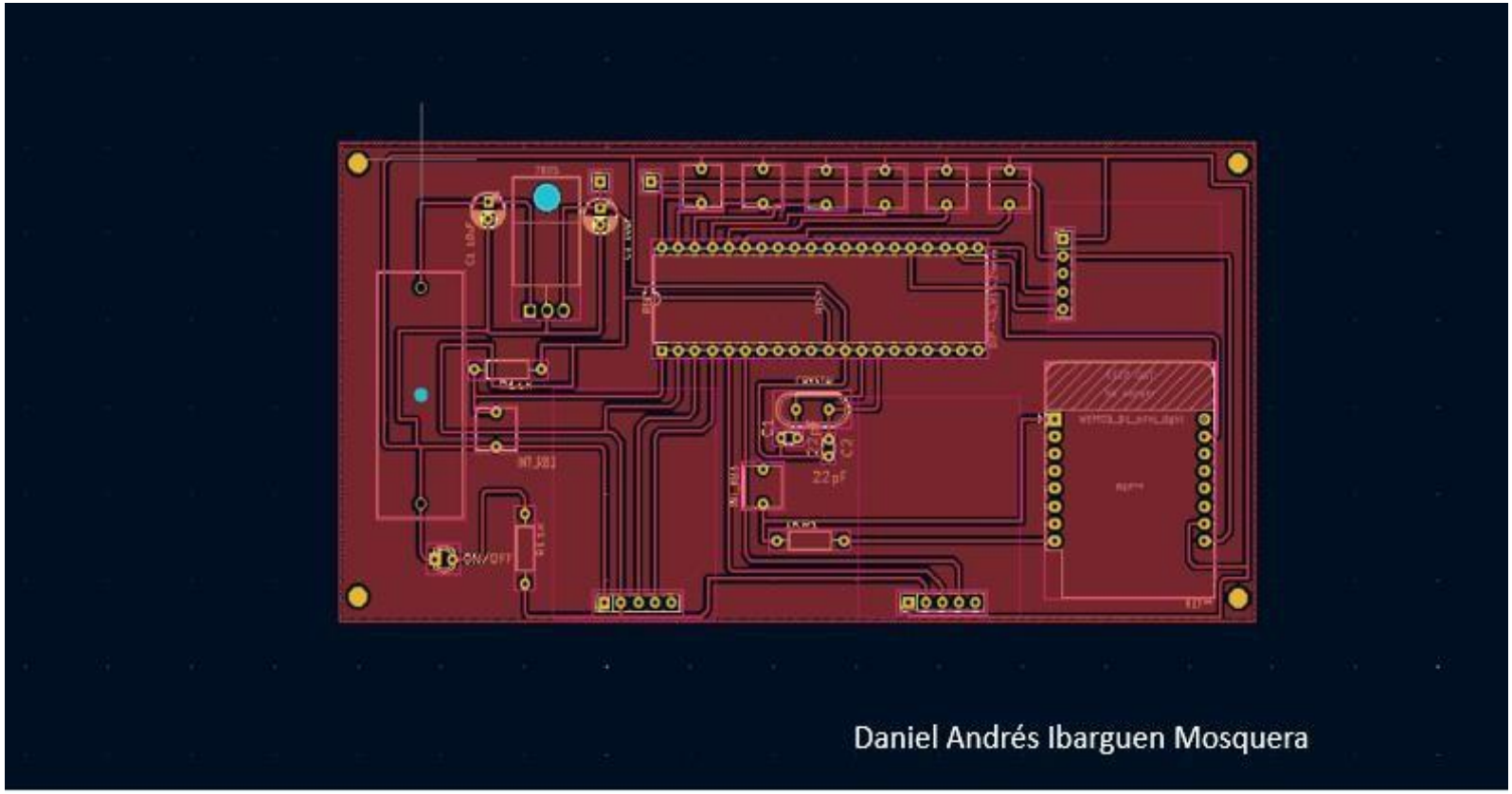
Fuente: Elaboración propia.

Diseño PCB Del Circuito Transmisor

En la figura 71 se puede observar el diseño PCB del circuito transmisor

Figura 71

PCB Circuito Transmisor



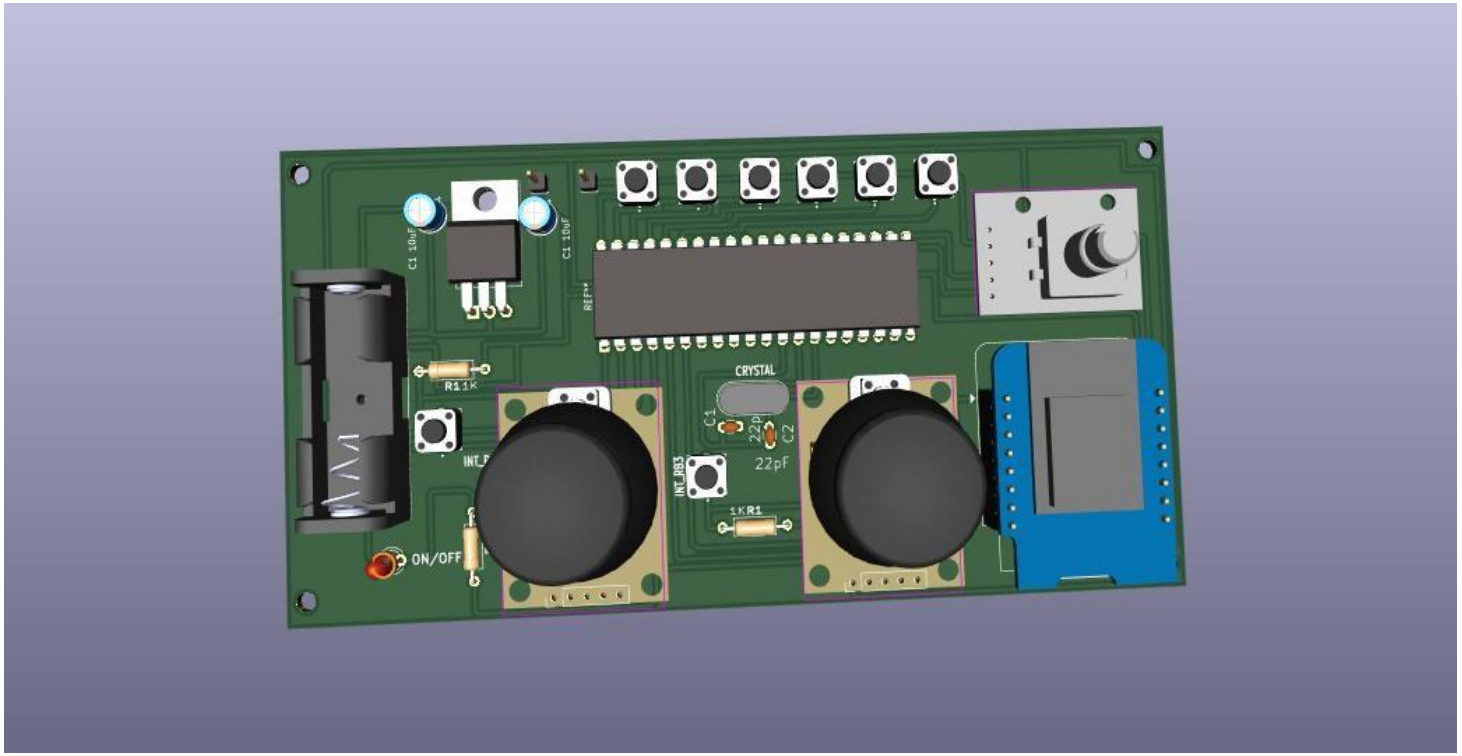
Fuente. Elaboración propia.

Modelo 3D Del Circuito PCB

En la figura 72 se puede observar el diseño 3D del circuito transmisor

Figura 72

Modelo 3D Control Remoto

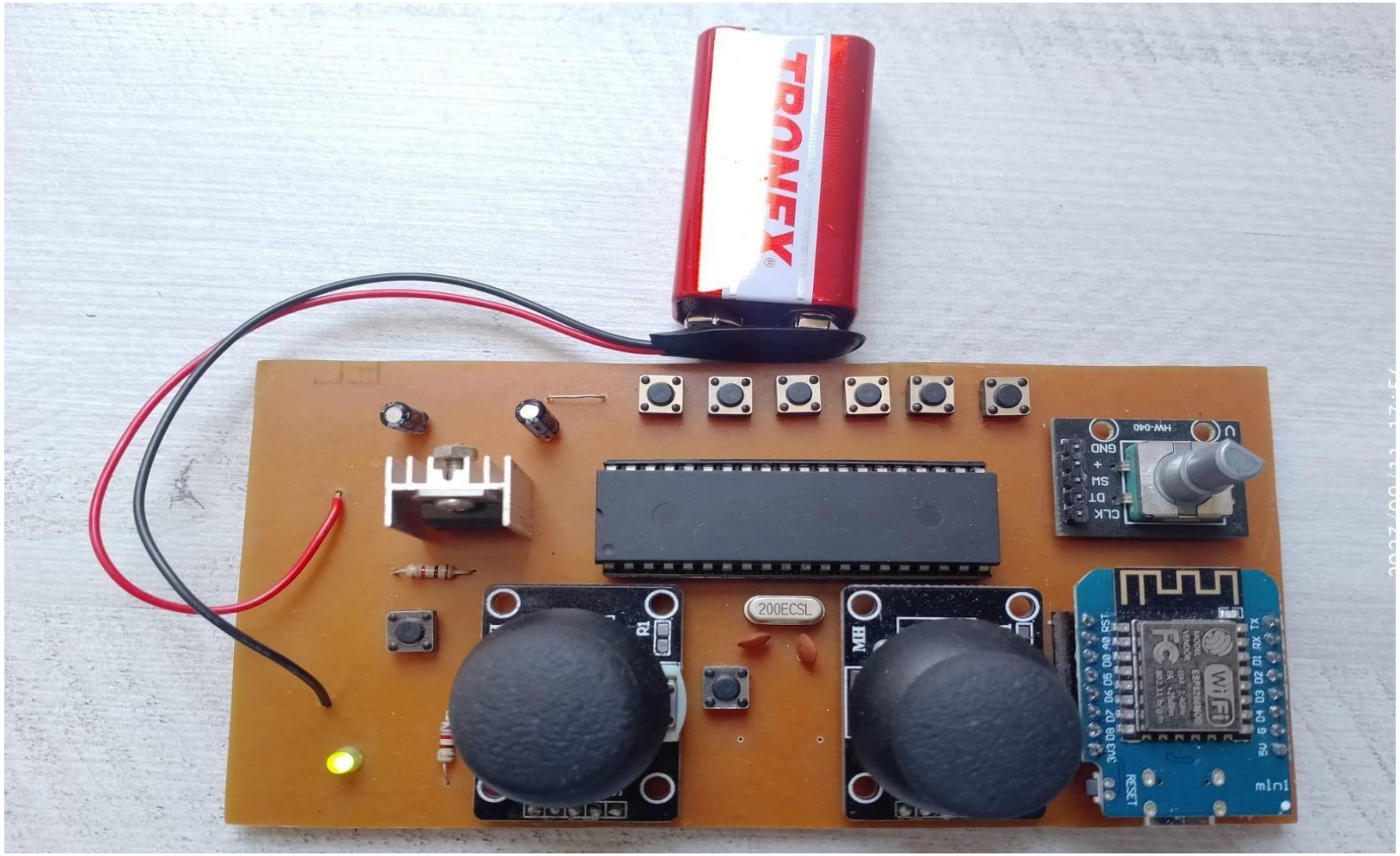


Fuente. Elaboración propia.

En la figura 73 podemos observar el circuito transmisor de forma física

Figura 73

Circuito Transmisor Físico



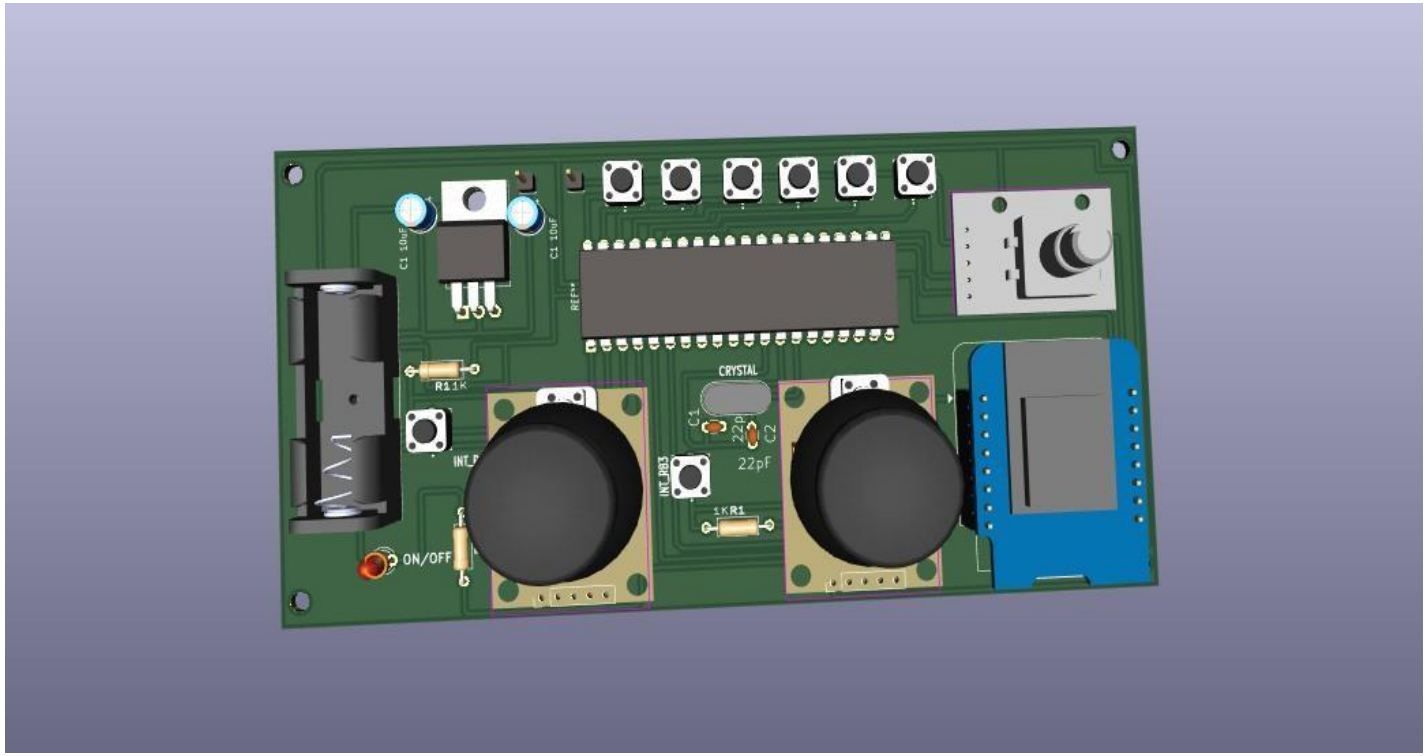
Fuente. Elaboración propia.

Especificaciones técnicas transmisor:

El control remoto transmisor de eventos se presenta en la figura 74:

Figura 74

Transmisor Inalámbrico



Fuente. Elaboración propia.

En la tabla 10 podemos observar las especificaciones técnicas del circuito transmisor

Tabla 10

Especificaciones Técnicas Hardware Transmisor De Eventos

Control Remoto Transmisor De Eventos: Especificaciones	
Alimentación	5V
Puerto USB	Tipo C
ADC Resolución	10 bits
Entradas digitales	5
Lecturas Encoder	3 x 2 Entradas digitales
Protocolo inalámbrico	EspNow
Comunicación Uart	9600, 38400, 115Baudios
Entrada de Voltaje	9V – 12V
Ancho	7Cm
Largo	14Cm
Banda de transmisión	2.4GHz
Alcance	10 a 50 metros
Driver	CH340 Windows 7, Windows 10

Nota. los datos corresponden a las características para el óptimo funcionamiento. *Fuente.* Elaboración propia.

Diseño del Módulo Receptor

En la última década hemos visto como ha crecido de manera exponencial el uso de aplicaciones inalámbricas permitiéndonos controlar dispositivos, maquinas o inclusive automatizar nuestra casa a través de aplicaciones con interfaces de conexión bluetooth o wifi. En esta sesión se explicará el desarrollo del módulo receptor. El receptor será encargado de recibir el buffer de datos proveniente del circuito transmisor anterior.

Como se describió en la sesión del circuito transmisor con el módulo ESP8266**,** mediante la interfaz de comunicación ESPNOW, desarrollada por la empresa ESPRESSIF, se ha diseñado un circuito básico que se encarga de suministrar la corriente necesaria para que un microcontrolador ESP8266 pueda operar en modo receptor. Para esto hemos utilizado adicionalmente tan solo una interfaz de comunicación USB, un circuito para la regulación de la fuente de voltaje proveniente de la propia conexión de la computadora y un circuito de programación para enviar el código del programa al ESP8266.

Para la implementación del módulo receptor se empleará el mismo chip ESP8266, pero en modo receptor el cual enviará los datos de control inalámbrico del usuario a la computadora. En la Figura 75 podemos ver el diagrama de bloques para cumplir este objetivo.

Figura 75

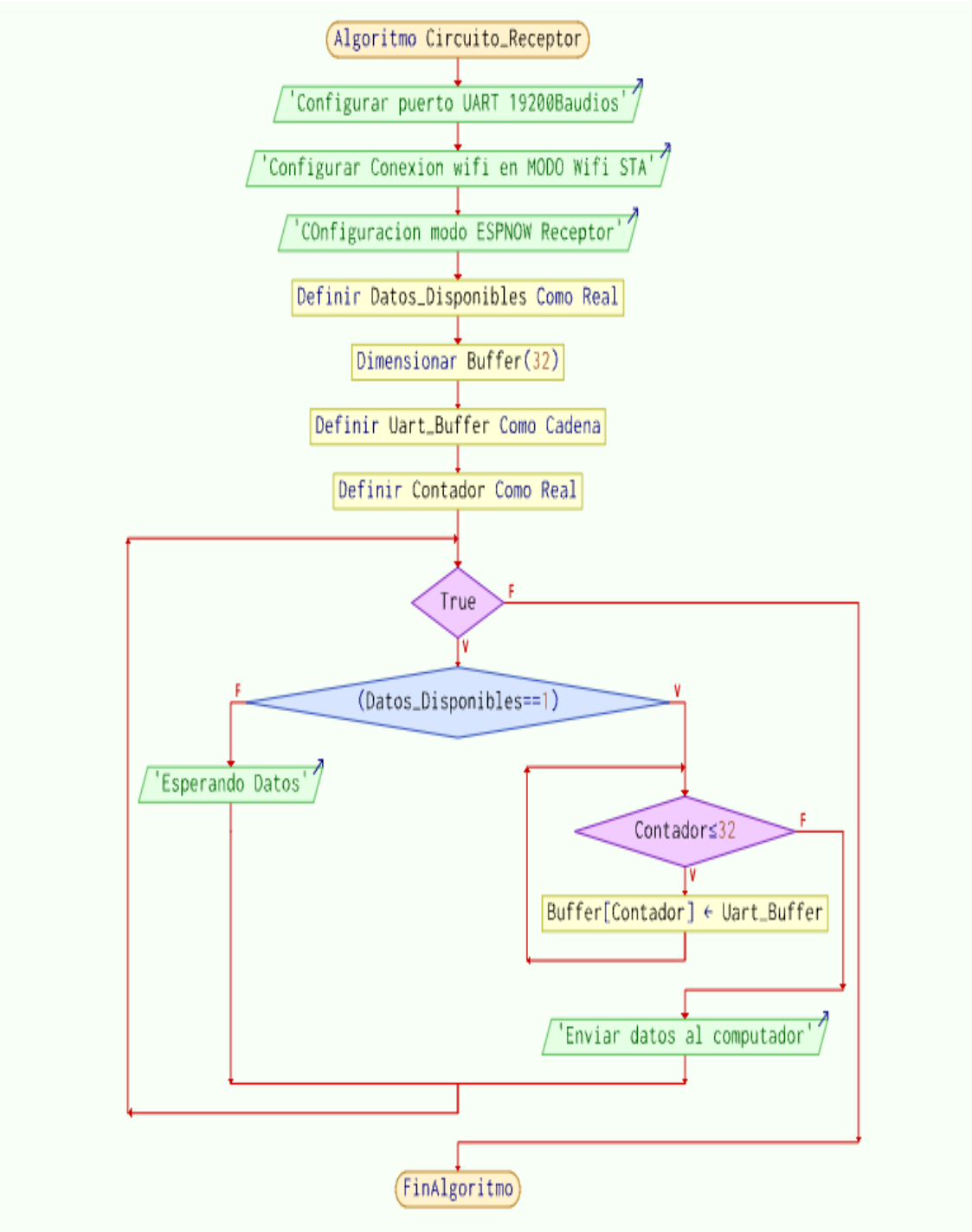
Diagrama de Bloques Metodología Objetivo Específico Numero 4

Fuente. Elaboración propia

El algoritmo que describe el funcionamiento interno del circuito receptor se puede observar en la figura 76, en donde se ha configurado previamente cada uno de los protocolos para dar paso al funcionamiento del circuito. Protocolo Usart, ESpnow, se debe tener en cuenta que la velocidad de funcionamiento del receptor en temas de procesamiento debe de ser mayor a la del circuito transmisor, para así reducir la perdida de datos, producto del ruido o interferencias naturales que se puedan presentar en el sistema.

Figura 76

Algoritmo Funcionamiento Circuito Receptor



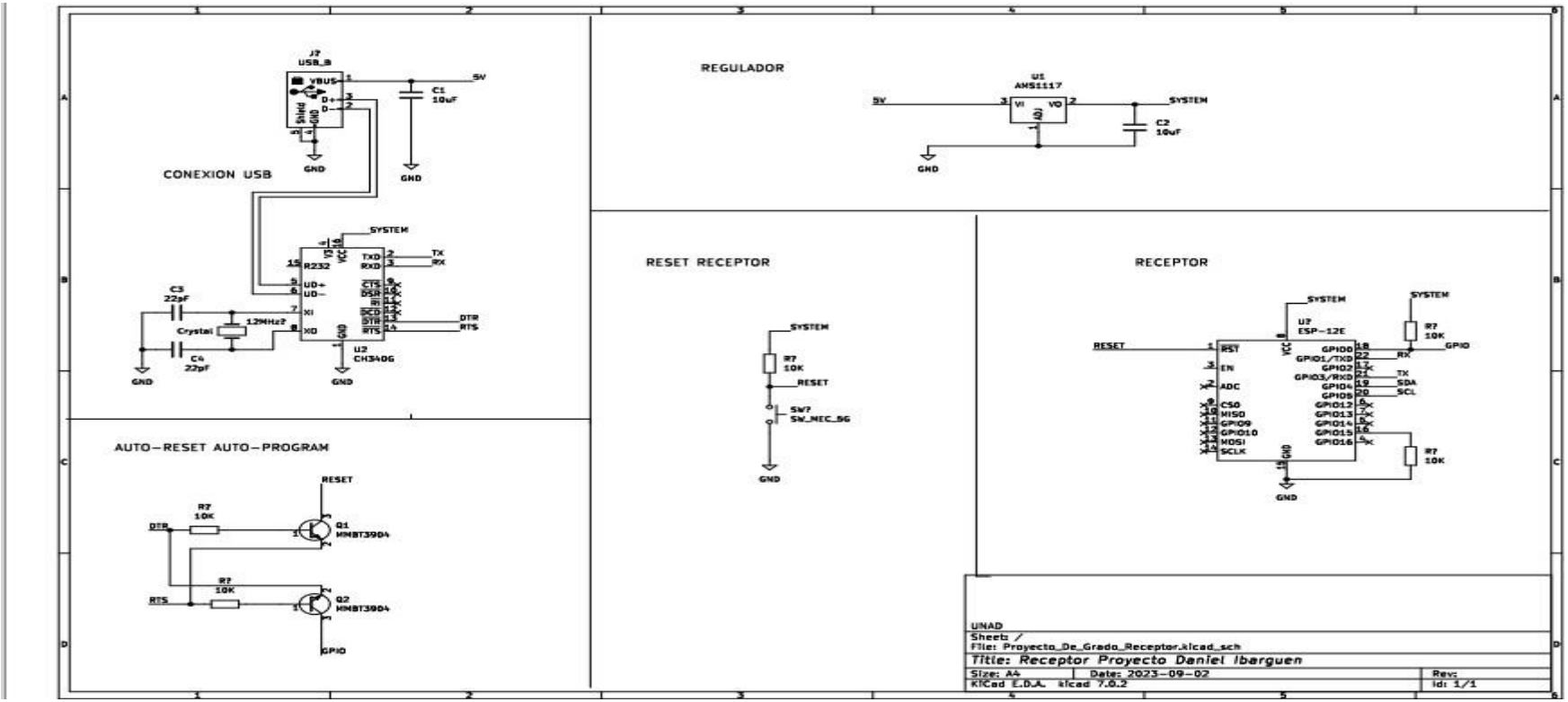
Fuente. Elaboración propia.

Circuito Receptor

En la figura 77 se puede observar el esquema electrónico final del circuito transmisor

Figura 77

Circuito Receptor



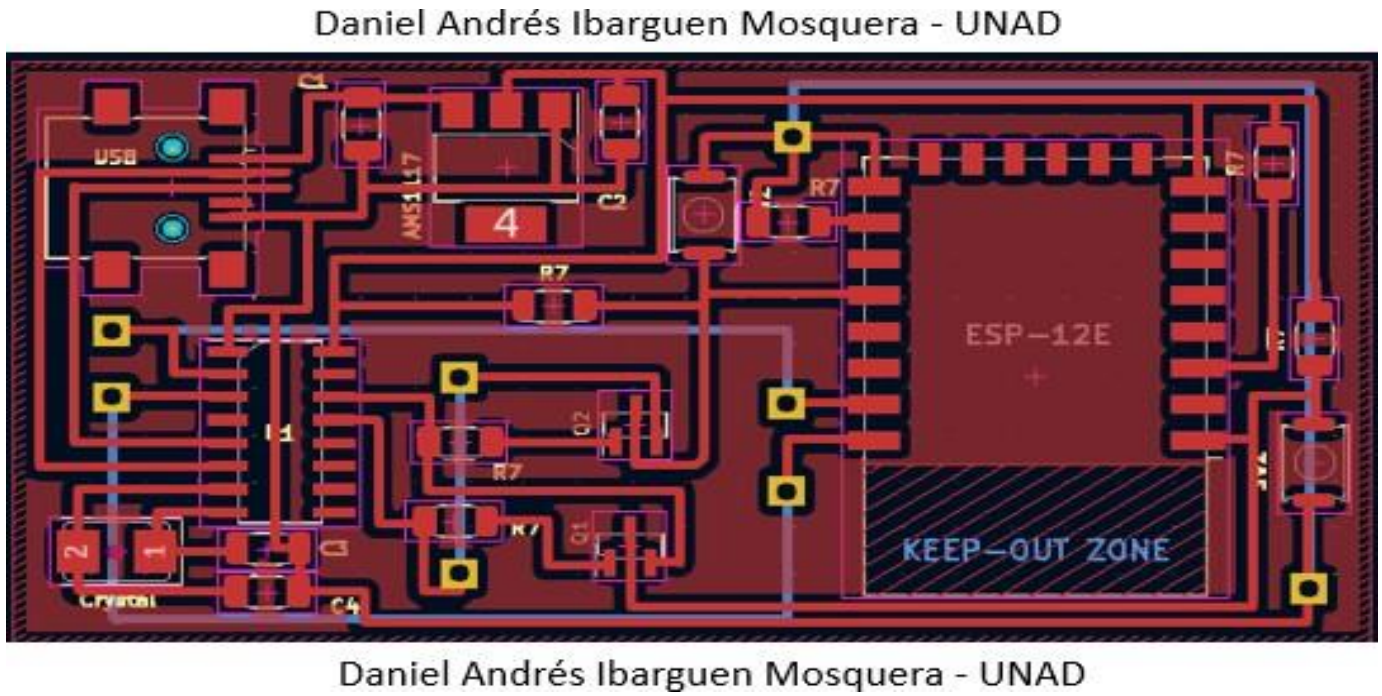
Fuente. Elaboración propia.

Esquema Circuito Receptor

En la figura 78 podemos observar el diseño PCB del circuito receptor

Figura 78

Esquema Kicad Circuito Receptor



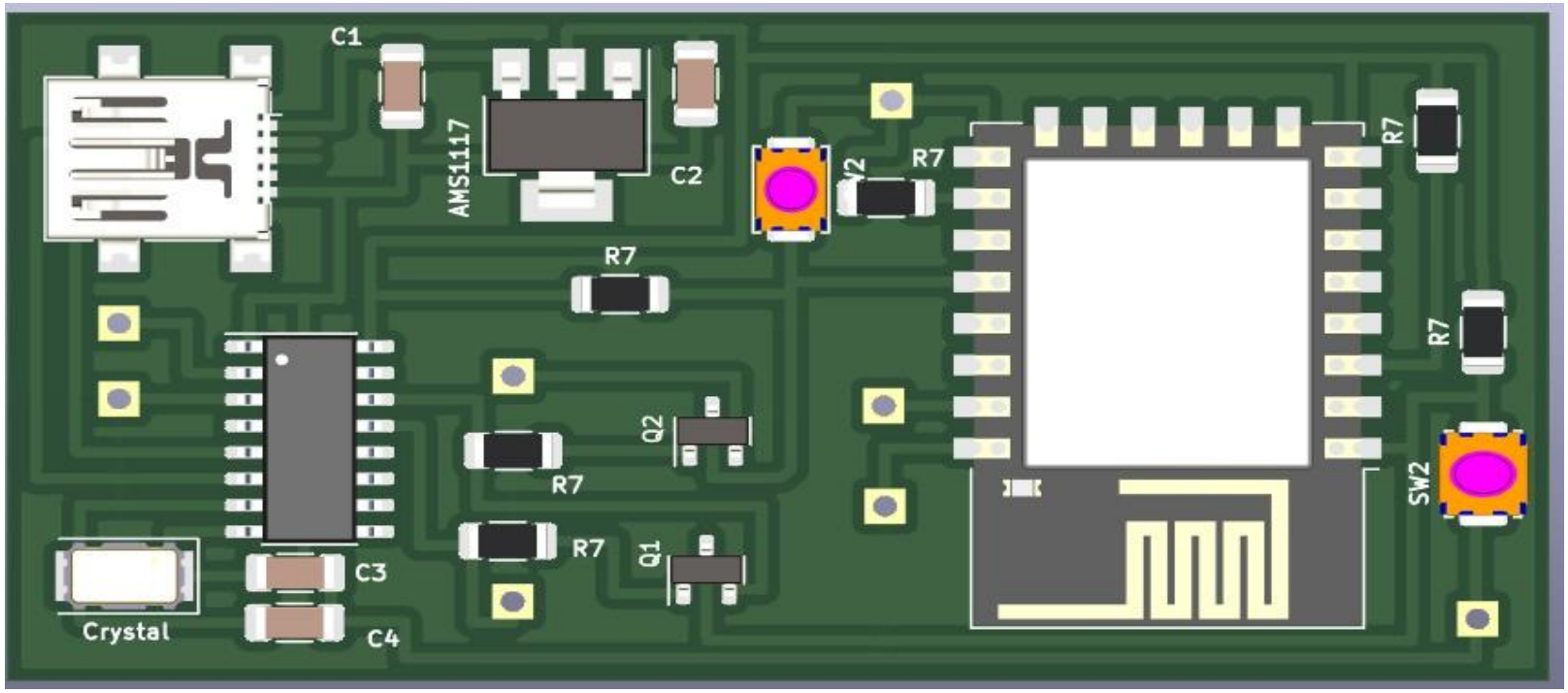
Fuente. Elaboración propia.

Modelo 3D del circuito Receptor

En la figura 79 se puede observar el diseño final del circuito receptor

Figura 79

Modelo 3D hecho en Kicad del Circuito Receptor



Fuente. Elaboración propia.

Especificaciones

En la tabla 9 podemos observar las características técnicas del hardware receptor

Tabla 11

Especificaciones Hardware receptor de eventos

Receptor De Eventos: Especificaciones	
Alimentación	5V
Puerto USB	Micro B
Protocolo inalámbrico	EspNow
Regulador Voltaje	3.3V
Ancho	3Cm
Largo	6,5Cm
Comunicación Uart	9600, 38400, 115Baudios
Banda de transmisión	2.4GHz
Alcance	10 a 50 metros
Driver	CH340 Windows 7, Windows 10

Nota. Descripción del hardware del circuito receptor.

Pruebas Finales e Información sobre el Software

En esta sección se mostrará los resultados finales del funcionamiento del sistema, el consumo de corriente del circuito transmisor y el consumo de recursos del software instalado en la computadora, de manera que el lector pueda entender el correcto uso y el funcionamiento de la aplicación.

Pruebas software del PC

Las pruebas que se realizan en el software permiten validar el correcto funcionamiento del sistema, detallando que el uso de Widgets como lo son: los botones, las barras de herramientas, el entorno gráfico y los datos en tiempo real del sistema sean igual a lo solicitado por el usuario. Se comienza por el empaquetado del programa utilizando la librería de Encapsulamiento de aplicaciones Pyinstaller.

Pyinstaller

Es una librería la cual le permite a un diseñador de programa escrito en Python obtener un archivo ejecutable el cual pueda correr en sistemas operativos como Linux y Windows. Su funcionamiento se basa en empaquetar todos los ficheros del programa en un solo archivo .exe El comando para realizar el empaquetamiento de ficheros con pyinstaller se puede observar en la figura 80.

Figura 80

Comando para Realizar un Programa Ejecutable Python

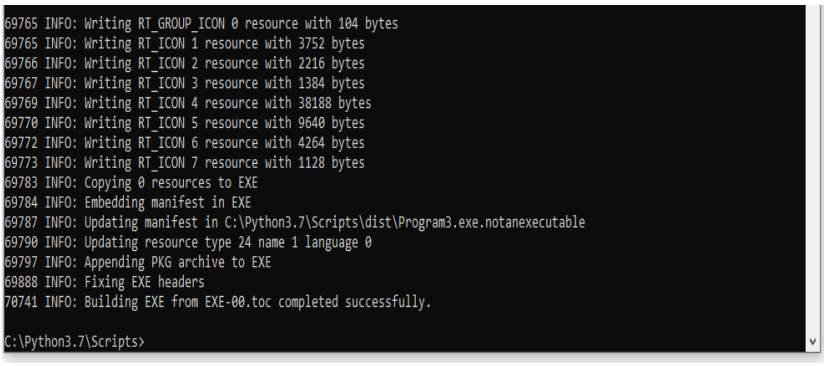
Fuente. Elaboración propia.

Una vez ejecutado el comando si el programa en Python no presenta errores en la compilación del código y en la ubicación de sus archivos, *Pyinstaller* retornara en el símbolo de

sistema el mensaje de finalizado Junto con un fichero. Exe de la aplicación, el resultado de estos procedimientos se puede observar en las figuras 81 y 82. En la última línea de la figura 81 podemos observar el resultado de satisfacción al terminar de empaquetar el código del programa y en la figura 82 podemos observar la carpeta con el archivo ejecutable para correr el programa.

Figura 81

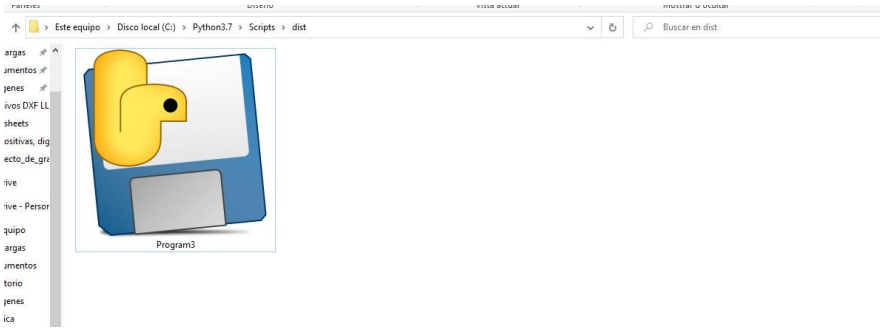
Resultado Empaquetamiento del Software



Fuente. Elaboración propia.

Figura 82

Programa de Simulación del Robot Empaquetado



Fuente. Elaboración propia.

Consumo de Recursos del Software de Computadora

Una vez empaquetado el software se procede a realizar su ejecución en el sistema operativo Windows, después de abierto se procede a revisar el administrador de tareas para observar el recurso de dicha aplicación como se puede observar en la figura 83

Figura 83

Consumo de Recursos Software de Simulación

Fuente. Elaboración propia.

Pruebas de Movimiento de la Cámara Mediante el Teclado.

Una vez ejecutado el software, la primera prueba que se realizará será utilizar el teclado para el movimiento de la cámara en el entorno 3D teniendo en cuenta que esta función también se puede realizar inalámbricamente mediante el control transmisor. Las teclas especiales son las siguientes:

W = Movimiento hacia adelante

S = Movimiento hacia atrás

A = Movimiento hacia la izquierda

D = Movimiento hacia la derecha

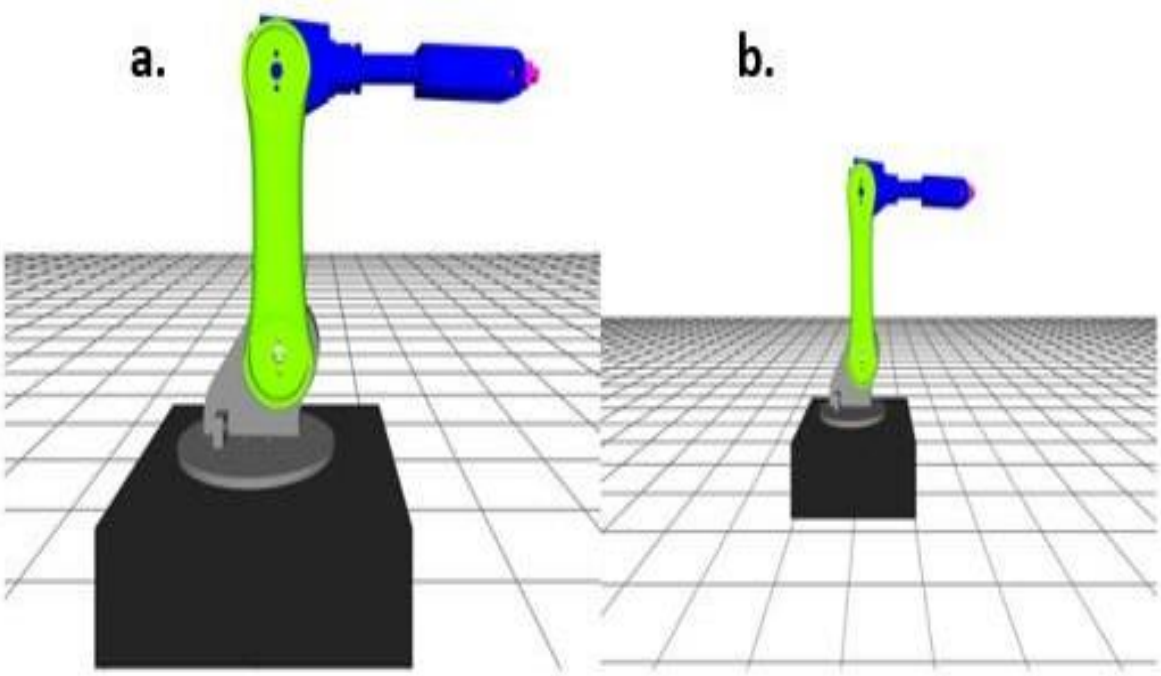
Q = Movimiento hacia arriba

E = Movimiento hacia abajo

En la figura 84 podemos apreciar los desplazamientos de la cámara hacia delante y hacia atrás

Figura 84

Desplazamiento hacia delante y hacia atrás Cámara



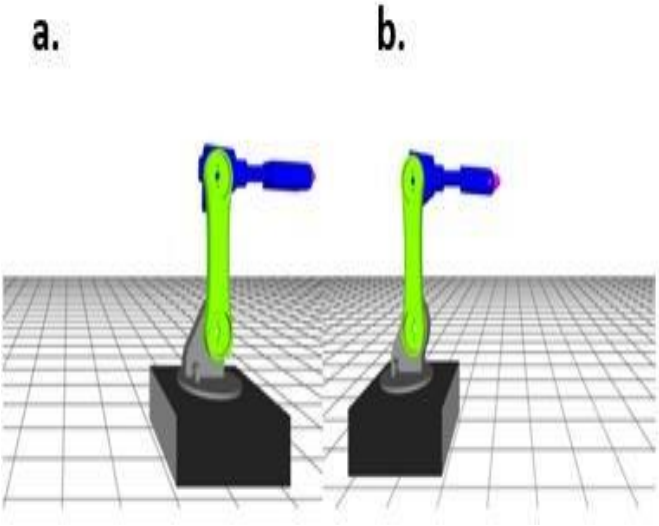
Nota. a. desplazamiento hacia delante b. desplazamiento hacia atrás

Fuente. Elaboración propia.

En la figura **85** podemos observar el movimiento hacia la izquierda y hacia la derecha

Figura 85

Movimientos Izquierda y Derecha Cámara



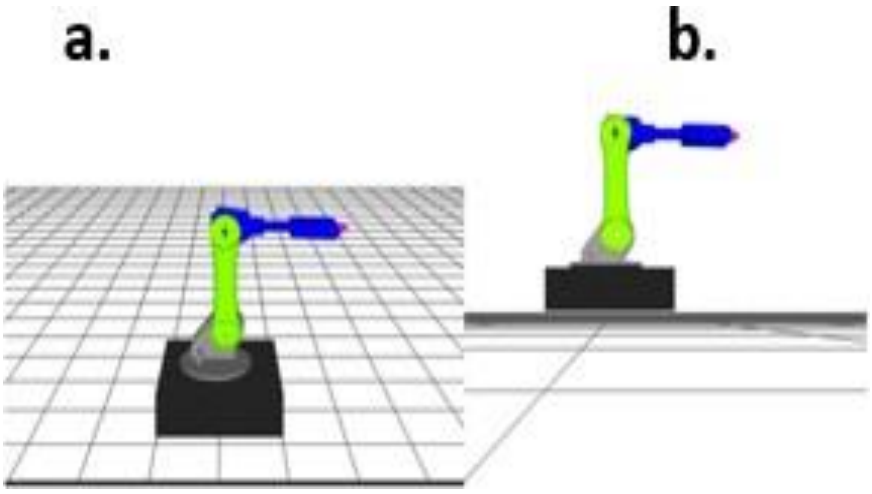
Nota. a. desplazamiento a la izquierda b. desplazamiento a la derecha

Fuente. Elaboración propia.

En la figura 86 y podemos observar el movimiento hacia arriba y hacia abajo.

Figura 86

Rotación Arriba y Abajo Cámara



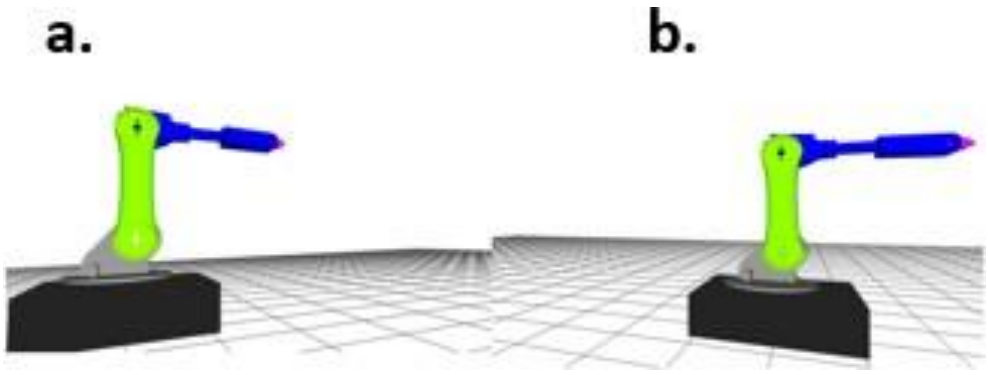
Fuente. Elaboración propia.

Pruebas de Rotación de la Cámara:

La cámara también tiene la posibilidad de ser rotada sobre su propio eje, esto con el fin de darle al usuario la sensación de que esta interactuando con un entorno 3D real. La cámara tiene la función especial de ser movida no solamente con el Control Transmisor, sino también con el ratón. Para realizar la prueba de rotación de la cámara, ésta se pondrá en un lugar fijo y luego se rotará sobre el eje Z hacia la izquierda y la derecha como podemos observar en la figura 87.

Figura 87

Rotación Izquierda y Derecha Cámara



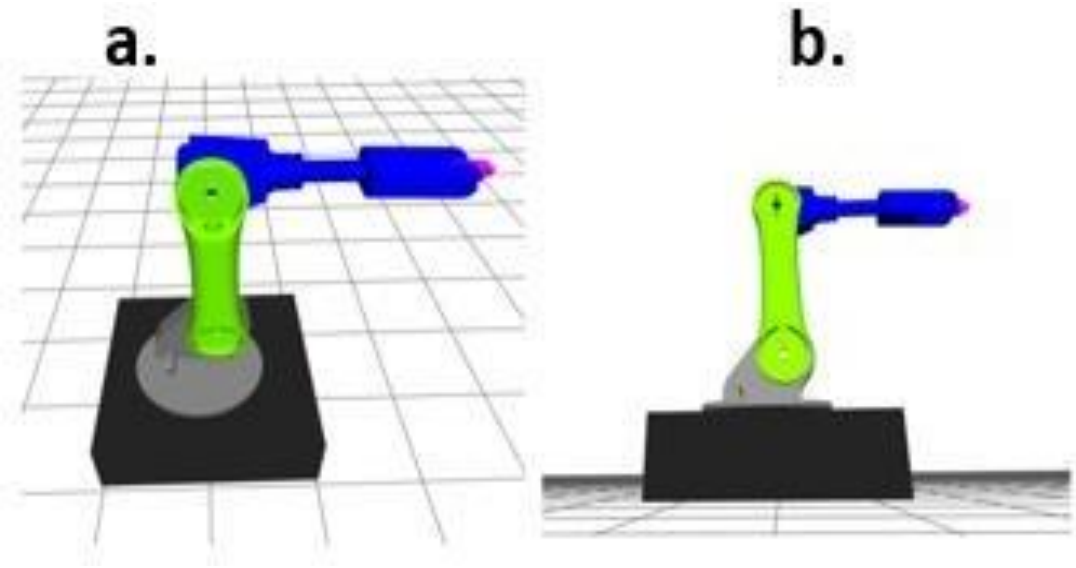
Nota. a. rotación cámara hacia la derecha b. movimiento cámara hacia la izquierda

Fuente: Elaboración propia.

Ahora se procede a realizar la rotación de la cámara en el eje Y en donde se verá orientaciones de vista de la cámara hacia arriba y hacia abajo como podemos observar en la figura 88.

Figura 88

Rotación Arriba y Abajo Cámara



Nota. a. rotación hacia abajo b. rotación hacia arriba. *Fuente.* Elaboración propia.

Prueba de Ventana Informativa Modelo del Brazo Robot NJT23

El software cuenta con una ventana que contiene información acerca del rango de operaciones del robot. Esta ventana se puede abrir en pantalla a través del menú de herramientas de la aplicación. Se procede a realizar su apertura y a realizar las pruebas en los botones que tienen la función de mostrar las diferentes imágenes. En la figura 89 podemos observar la barra de herramientas de la interfaz, una vez situados en el texto de ficha técnica se mostrará el nombre del brazo robot actual. Al hacer click en NJT23 se mostrará la ventana informativa de la figura 90.

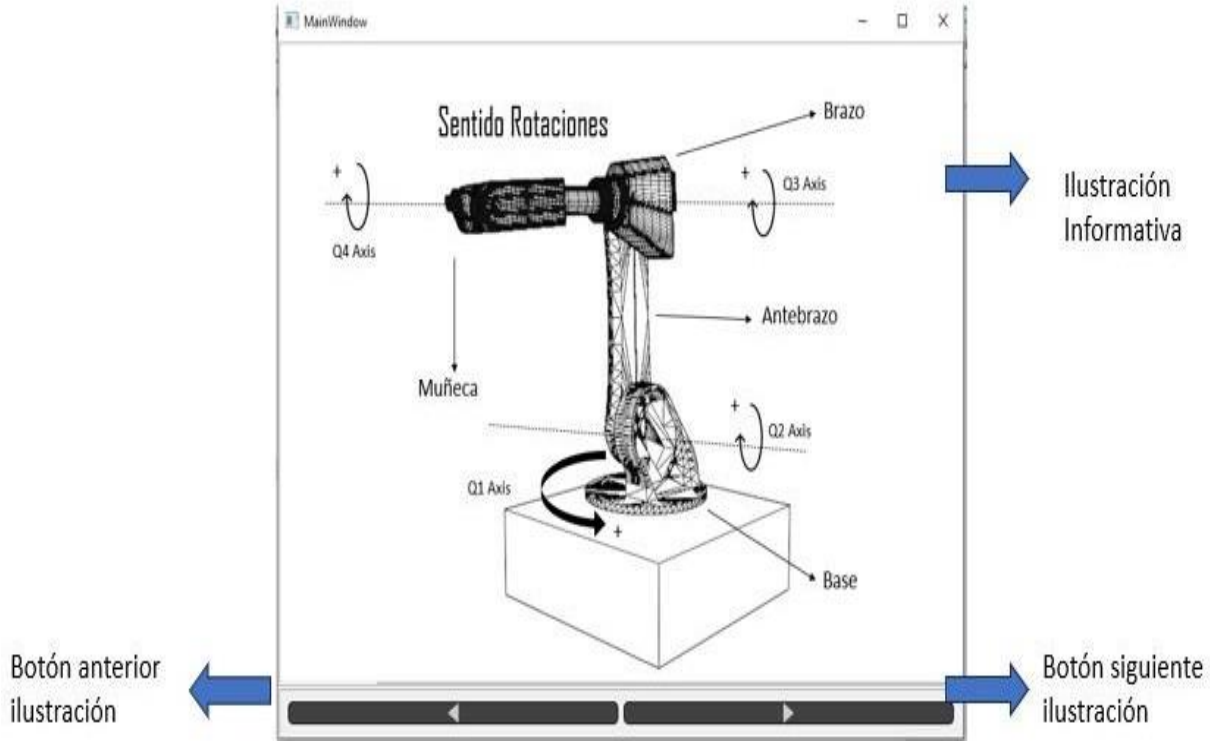
Figura 89

Menú de Herramientas Software NJT23

Fuente. Elaboración propia.

Figura 90

Ventana Informativa Datos Robot NJT23

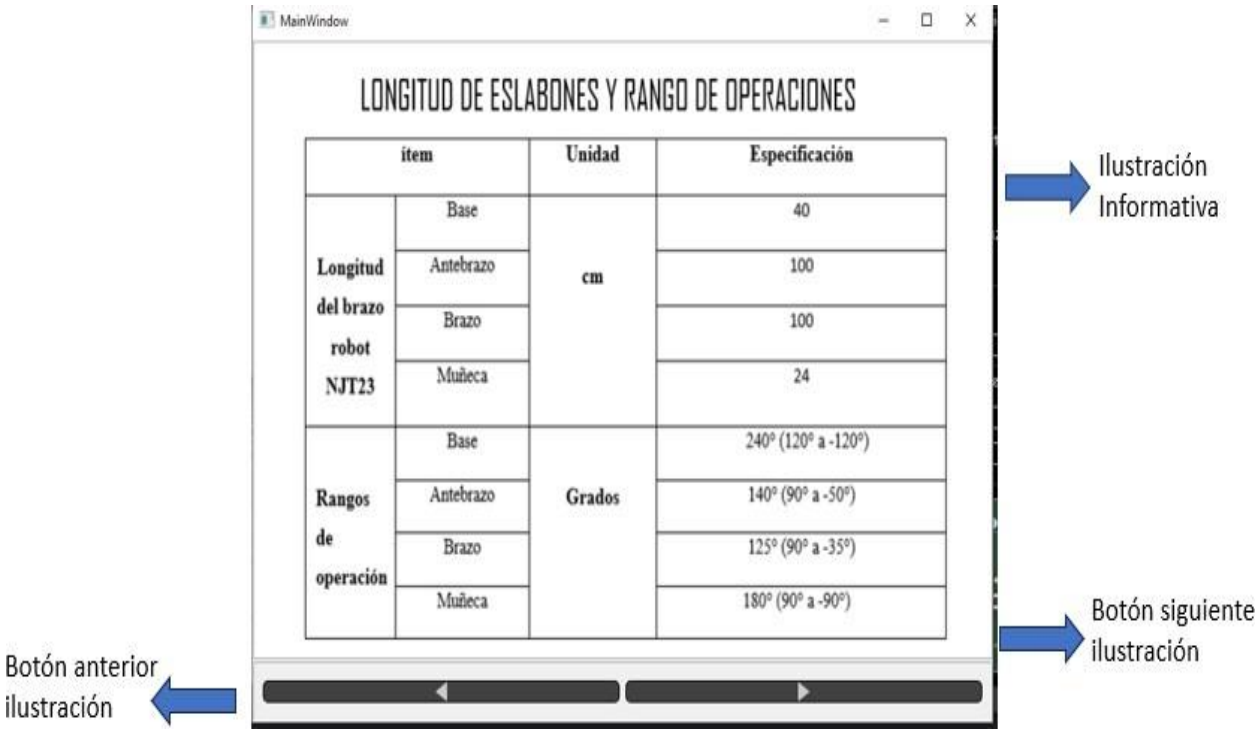


Fuente. Elaboración propia.

Una vez abierta la ventana informativa se procede a dar clic en los botones para mostrar las siguientes ilustraciones con los datos de rotaciones, longitudes de eslabones e identificadores como se puede observar en las figuras 91 y 92.

Figura 91

Información Longitud Eslabones y Rango Rotaciones

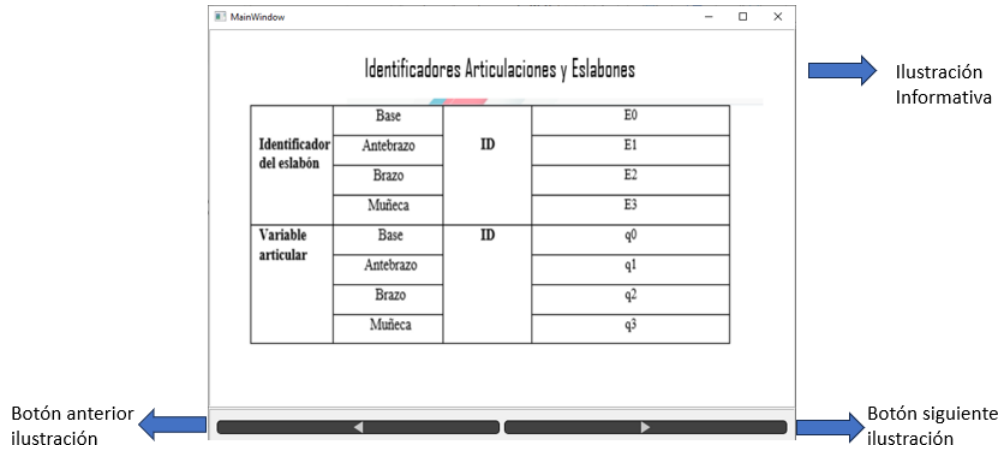


ítem		Unidad	Especificación
Longitud del brazo robot NJT23	Base	cm	40
	Antebrazo		100
	Brazo		100
	Muñeca		24
Rangos de operación	Base	Grados	240° (120° a -120°)
	Antebrazo		140° (90° a -50°)
	Brazo		125° (90° a -35°)
	Muñeca		180° (90° a -90°)

Fuente. Elaboración propia.

Figura 92

Ilustraciones Informativa Identificadores de Eslabones y Articulaciones



Fuente. Elaboración propia.

Prueba de Funcionamiento Widgets Informativos en Tiempo Real del Programa

Como se mencionó en capítulos anteriores el software cuenta con una serie de widgets que muestran las siguientes informaciones en tiempo real.

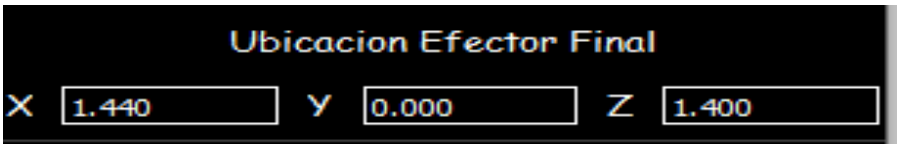
- Posición actual del efector final
- Valores de Angulo de rotación de cada articulación
- Estado del programa.

Posición del Efector Final

El software tiene la capacidad de dar al usuario en tiempo real la posición en metros del efector final. En la figura 93 se puede apreciar el valor actual del efector final cuando no hemos conectado el control transmisor y cuando todos los valores de rotación de cada articulación del robot se encuentran en 0. En el próximo capitulo se dará a conocer el funcionamiento del widget con la conexión del control remoto de manipulación del robot.

Figura 93

Widget Informativo Ubicación Efector Final



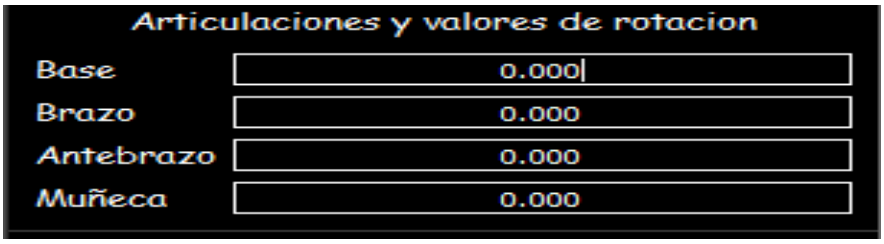
Fuente. Autor del proyecto.

Valores de Angulo de Rotación de cada Articulación

El widget presentado en la figura 94 muestra al usuario el valor de rotación actual de cada articulación del robot. Trabaja en conjunto con el widget anterior para el cálculo de la posición actual del efector final. Al iniciar el programa y no obtener ningún valor de rotación podemos observar que todos los valores de rotación se encuentran en 0.

Figura 94

Valor de Rotación de cada Articulación



Fuente. Elaboración propia.

Widget Estado del Programa.

Aquí podemos apreciar información sobre el número de fotogramas por segundo el cual corre el programa, el estado de conexión del transmisor y el receptor y la cantidad de datos que se han enviado a la GPU, como se muestra en la figura 95.

Figura 95

Widget Informativo Estado del Software

Fuente. Elaboración propia.

Conexión y Prueba del Circuito o Modulo Receptor

El software se comunica con el control remoto a través de un circuito receptor como se ha mostrado en capítulos anteriores. Esa conexión se realiza mediante una interfaz USART también descrita anteriormente. En esta sesión se comprobará que el programa sea capaz de conectarse con el módulo de recepción de datos.

Pasos para Verificación de Conexión del Receptor:

1 - Conectar el módulo receptor a algún puerto USB de la computadora como se puede observar en la figura 96.

Figura 96

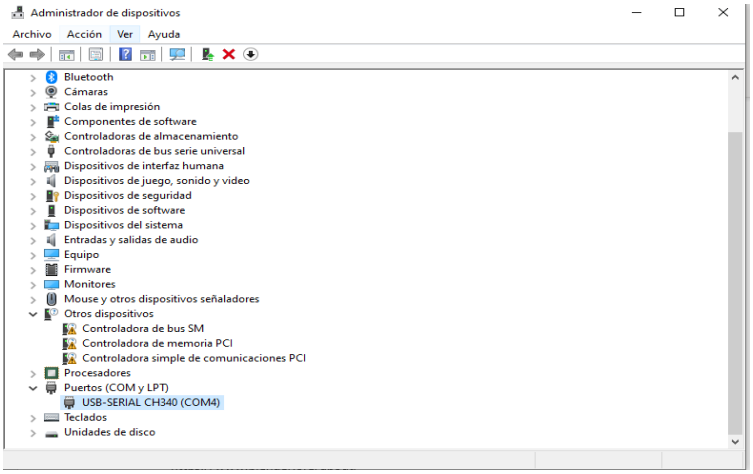
Conexión Modulo Receptor al PC

Fuente. Elaboración propia.

2 -Entrar al administrador de dispositivos del sistema, desplegar la opción Puertos COM y LPT y mirar el número del puerto al cual el dispositivo ha sido asignado como se puede ver en la figura 97.

Figura 97

Verificación Conexión Puerto COM Receptor

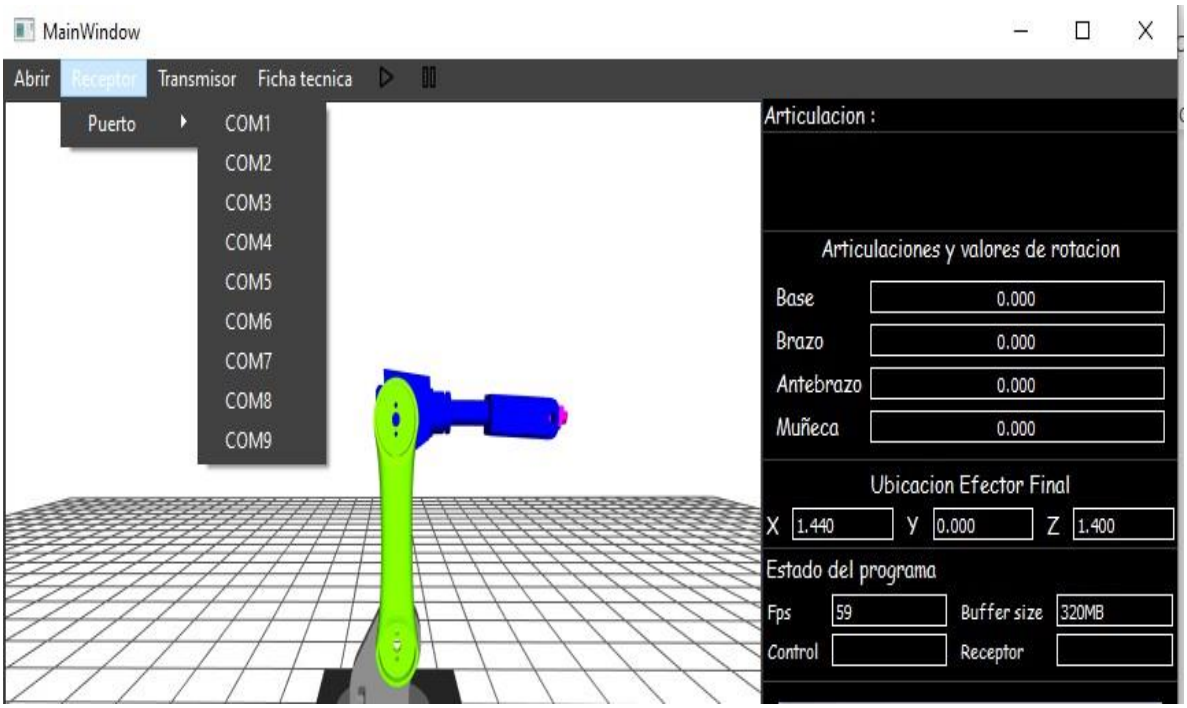


Fuente. Elaboración propia.

3 - Dirigirse al menú de herramientas del software, seleccionar la opción Receptor – Puerto y luego elegir el puerto COM que se encuentra en la computadora como se puede observar en la figura 98.

Figura 98

Selección del Puerto COM del Receptor

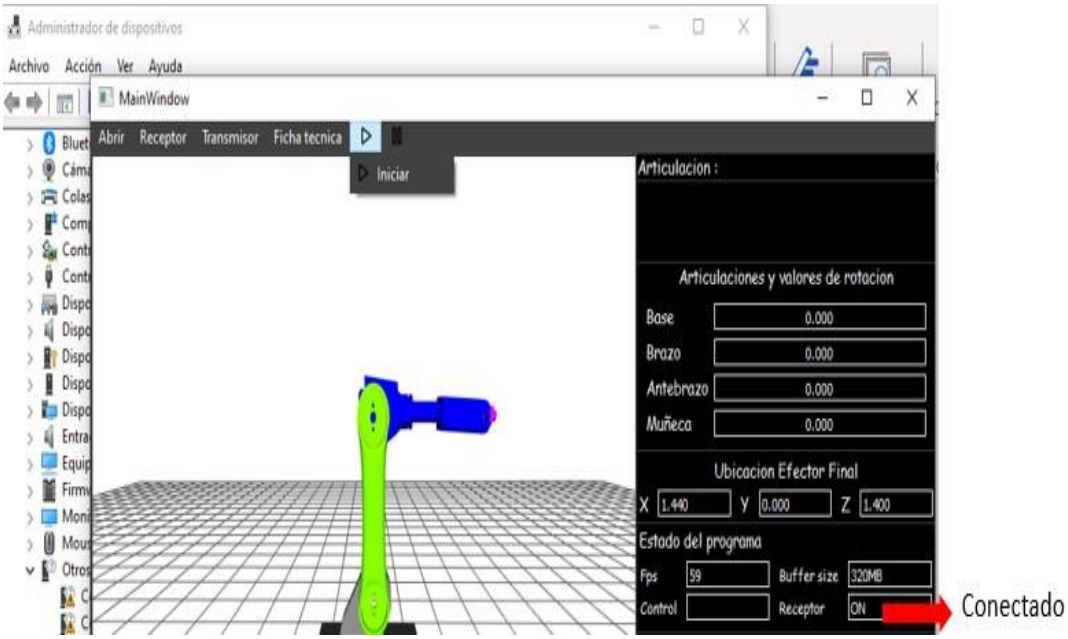


Fuente. Elaboración propia.

4 - Dirigirse al menú de herramientas símbolo de Play o inicio y seleccionar iniciar. Si hay algún error en la conexión del receptor se mostrara el mensaje de NO CONNECT en el widget de información de estado del programa. En caso de que la conexión sea exitosa se podrá ver el mensaje ON como se muestra en la figura 99.

Figura 99

Verificación de Conexión con el Receptor



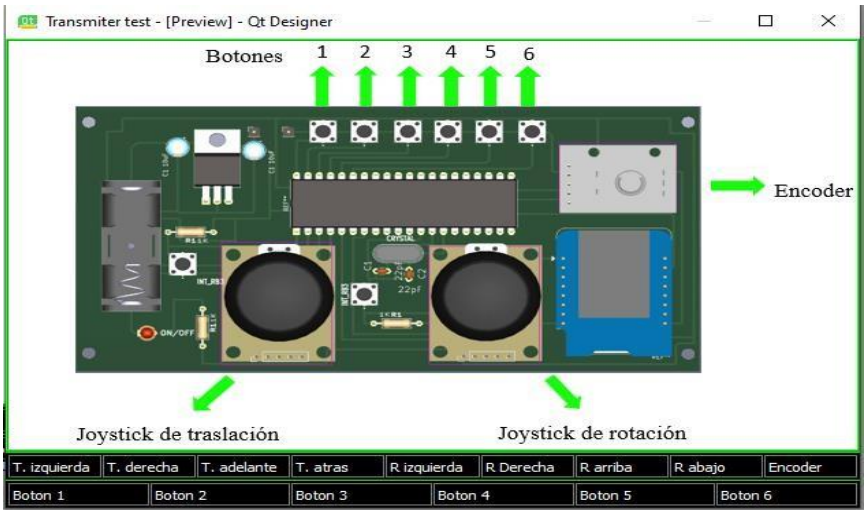
Fuente. Elaboración propia.

Conexión y Prueba del Circuito Transmisor

Para las pruebas del circuito transmisor se ha diseñado una ventana de pruebas del control remoto en el software del robot NJT23 como se puede observar en la figura 100. En este ventana el usuario tendrá la posibilidad de observar el funcionamiento de cada uno de los periféricos que conforman el circuito transmisor y también el estado y la transmisión de datos por parte del receptor.

Figura 100

Ventana de Pruebas del Control Remoto

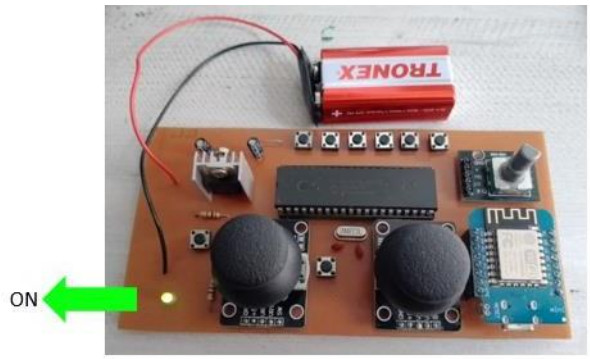


Fuente. Elaboración propia.

Al conectar la batería en el control remoto el led de indicación de encendido alumbra como se muestra en la figura 101.

Figura 101

Indicación Control Remoto Encendido



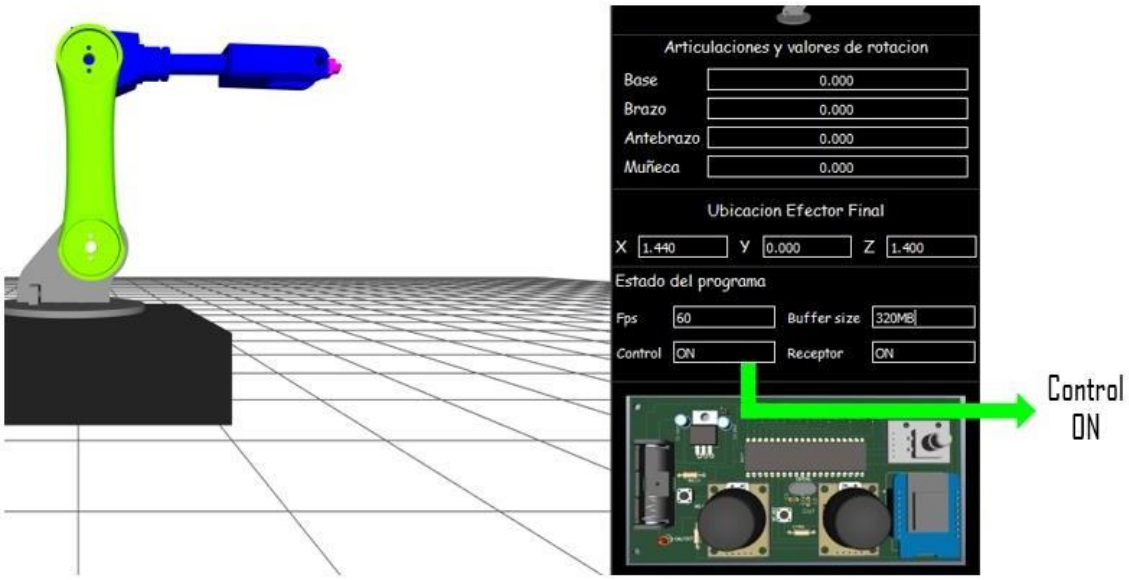
Fuente. Elaboración propia.

Para acceder a la ventana de pruebas de datos del control remoto es necesario dirigirse al menú de herramientas de la interfaz y realizar la conexión del módulo receptor como se explicó

anteriormente. Si el control remoto está conectado se mostrará el estado del control On como se observar en la figura 102.

Figura 102

Control Remoto ON

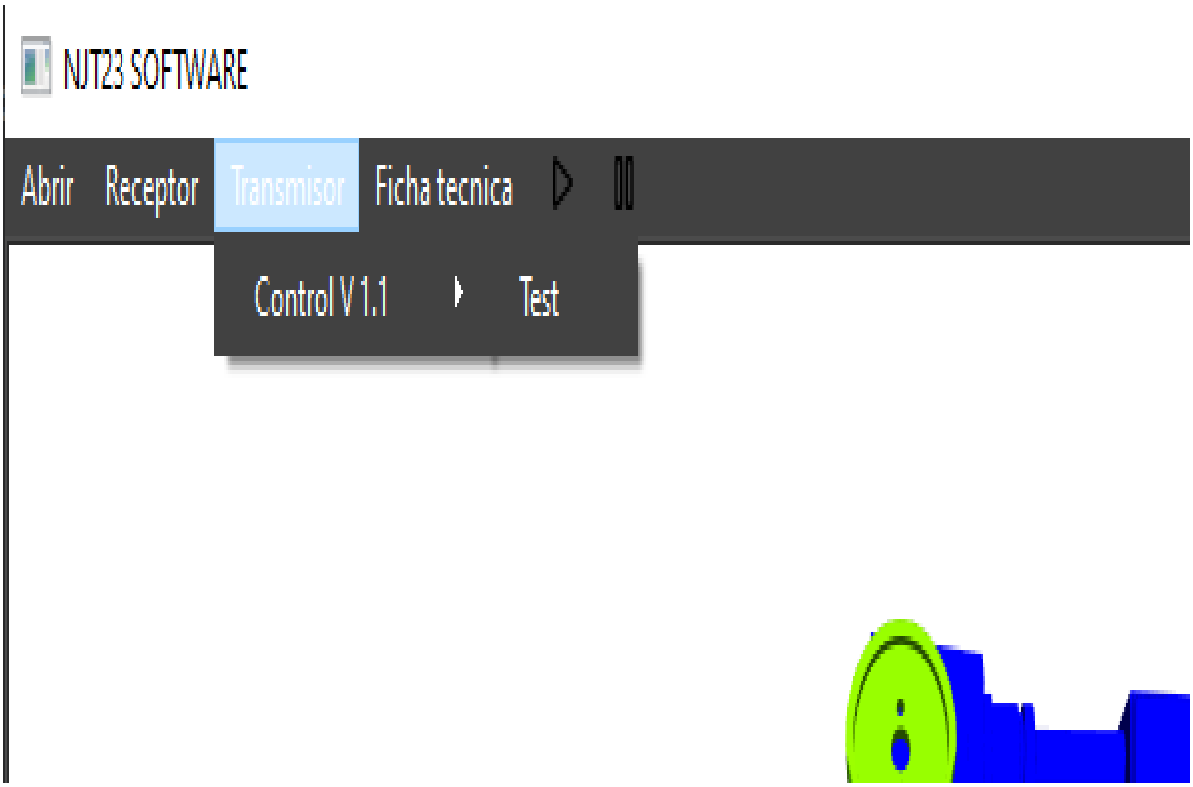


Fuente. Elaboración propia.

En el Widget de información del programa desplegar la opción Transmisor – Control V 1.1 – test como se observa en la figura 103.

Figura 103

Apertura Ventana de Pruebas



Fuente. Elaboración propia.

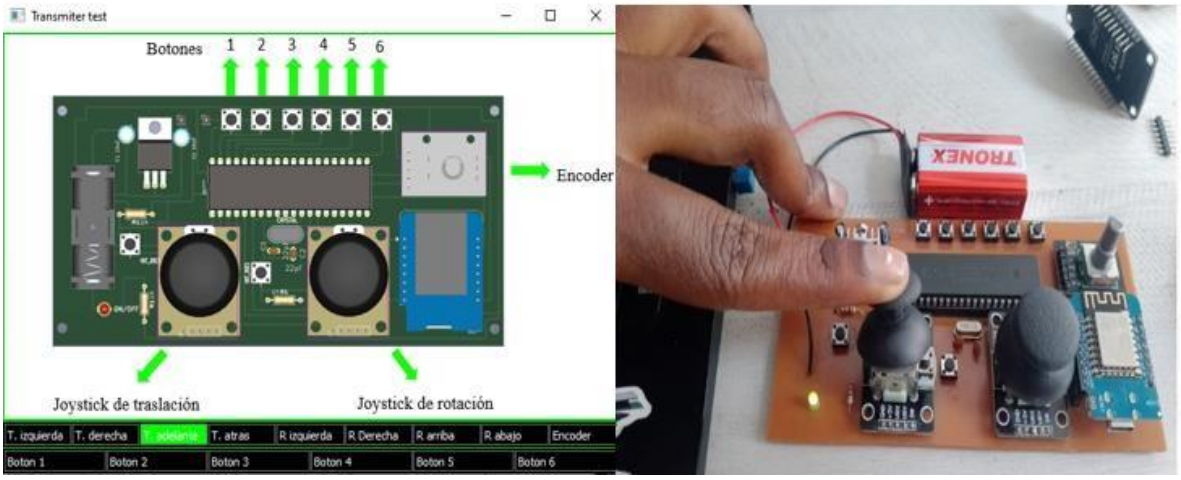
Una vez abierta la ventana se puede empezar a interactuar con cada uno de los diferentes periféricos del control retomo y analizar los cambios en la interfaz.

Prueba Joystick de Traslación de Cámara

Al mover el joystick hacia adelante o hacia atrás podemos observar los cambios de estado en forma de color en las etiquetas de la pantalla como se puede observar en las figuras 104 y 105.

Figura 104

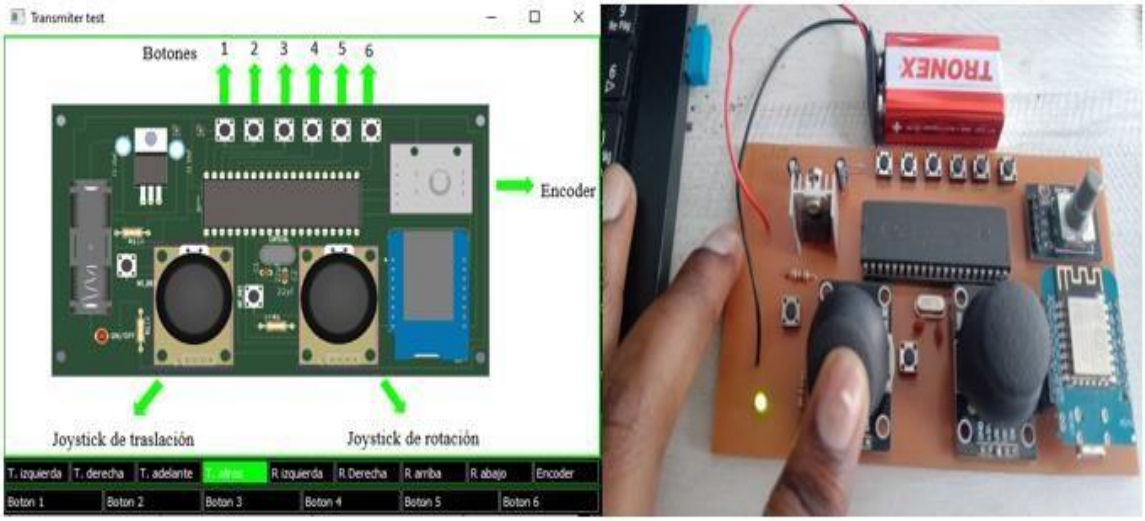
Cambio de Estado Mover Joystick Hacia Delante



Fuente. Elaboración propia.

Figura 105

Cambio de Estado Mover Joystick hacia atrás

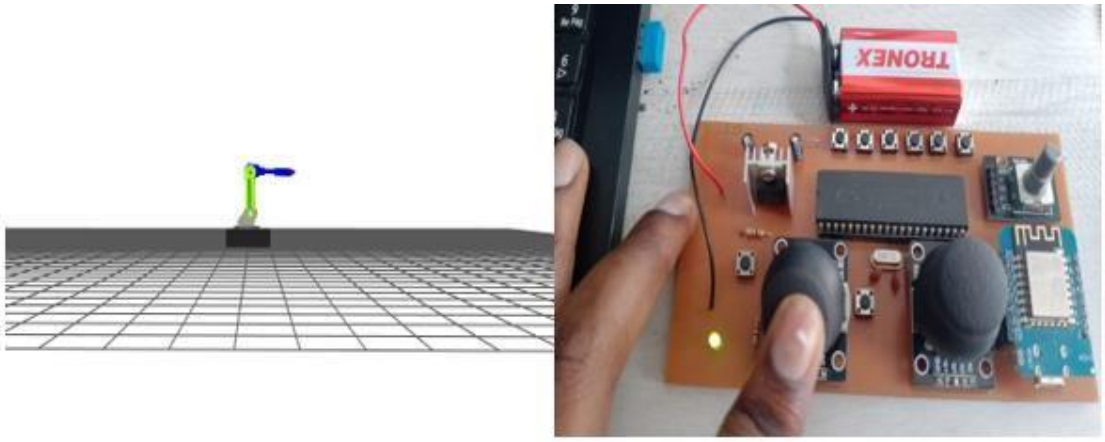


Fuente. Autor del proyecto

De la misma manera si cerramos la ventana y regresamos a la ventana principal de movimiento del robot podemos cambiar la posición de la cámara como se muestra en la figura 106 y figura 107.

Figura 106

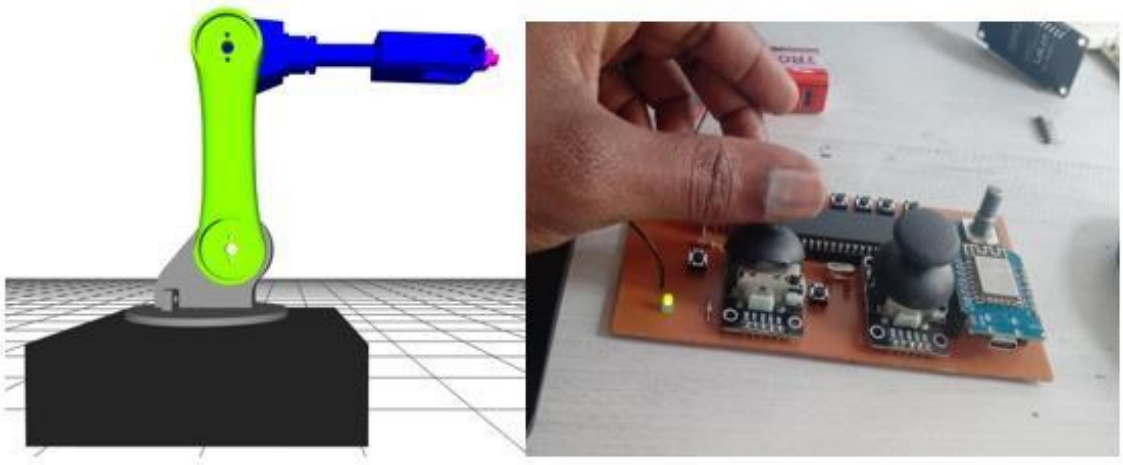
Cámara Alejada con el Joystick



Fuente. Elaboración propia.

Figura 107

Acercamiento de Cámara con Joystick



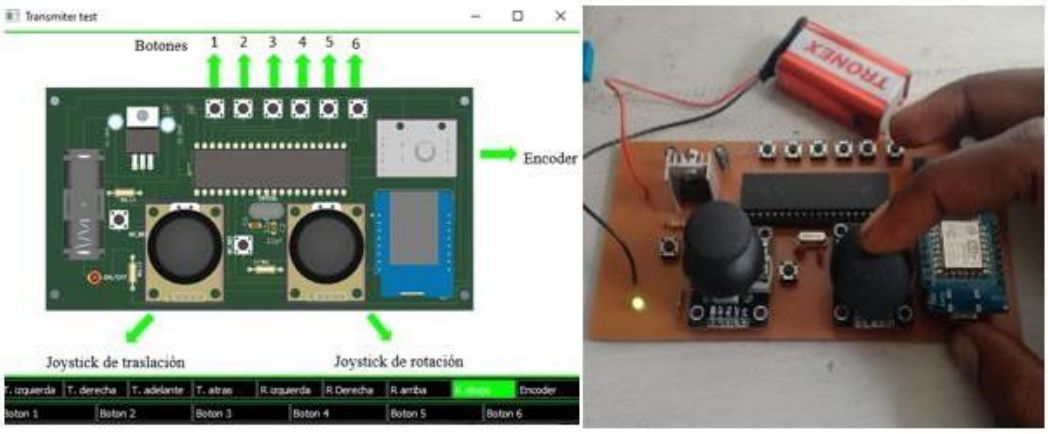
Fuente. Elaboración propia.

Prueba del Joystick de Rotación de Cámara

Para realizar la prueba en la rotación de la cámara se ha movido el joystick derecho hacia arriba y hacia abajo y se han analizado los cambios en la ventana de pruebas del control como podemos observar en las figuras 108 y 109.

Figura 108

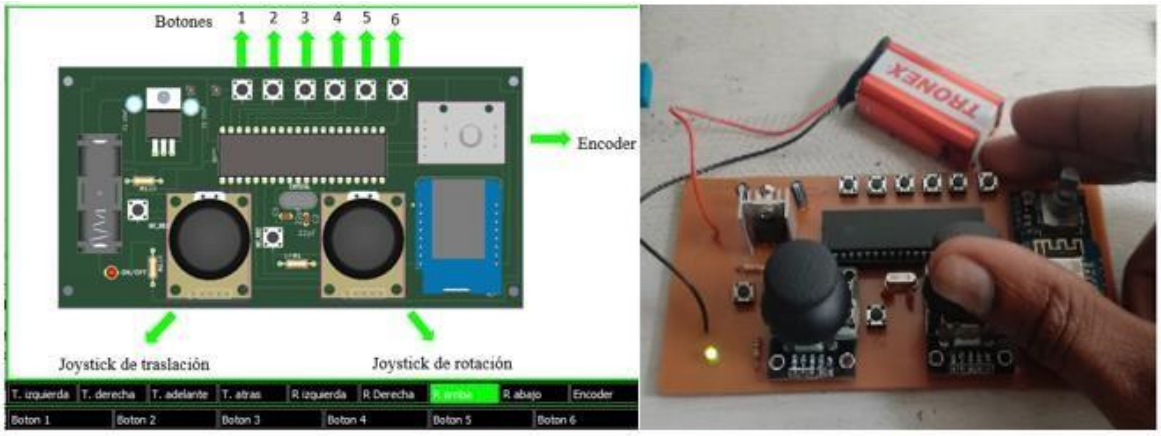
Movimiento del Joystick de Rotación Hacia Abajo



Fuente. Elaboración propia.

Figura 109

Movimiento Joystick de Rotación Hacia Arriba

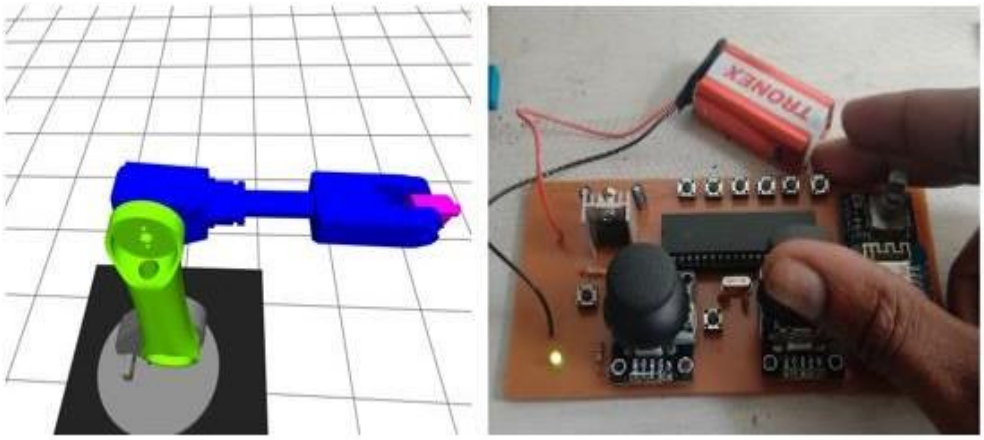


Fuente. Elaboración propia.

En las figuras 110 y 111 podemos observar los cambios de rotación de la cámara de acuerdo con el joystick de rotación desde una posición fija desde arriba del robot.

Figura 110

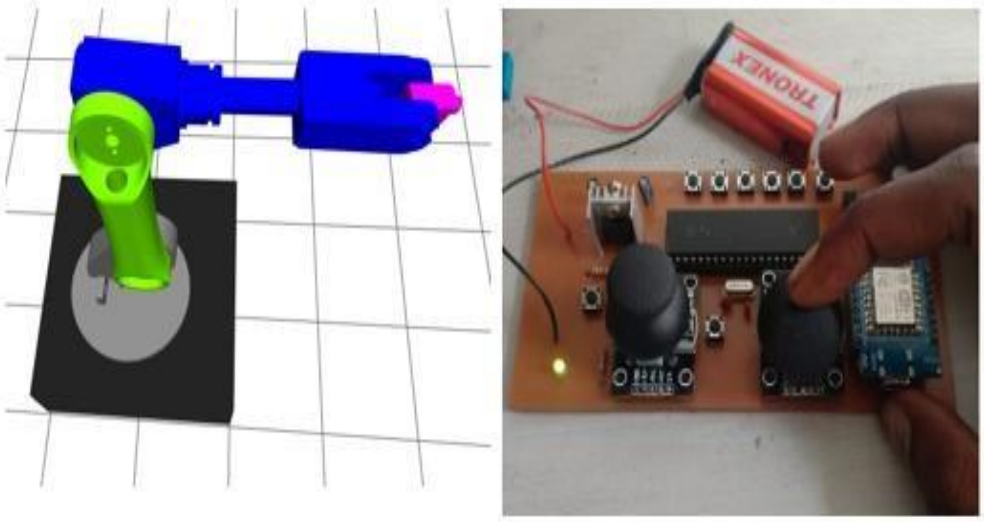
Rotación de la Cámara Hacia Arriba



Fuente. Elaboración propia.

Figura 111

Rotación de la Cámara Hacia Abajo



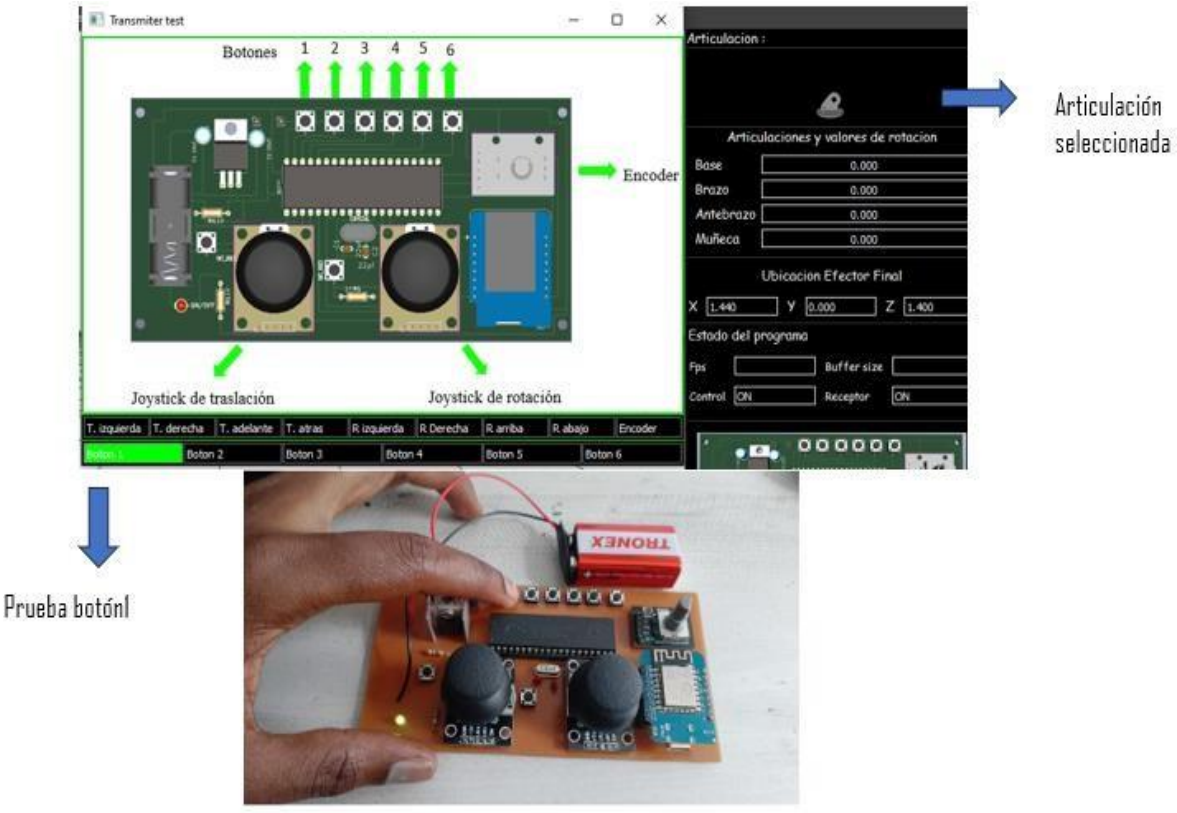
Fuente. Elaboración propia.

Prueba de Botones y Selección de Articulación

Los botones ubicados en la parte superior del control remoto nos sirven para seleccionar la articulación que se desea rotar, también para restablecer el robot a su posición inicial o si se tiene algún problema con la orientación de la Cámara, poder restablecerla a su valor inicial. Para la prueba se ha utilizado igualmente la ventana de pruebas del robot y al realizar las pulsaciones en cada uno de ellos se puede observar los cambios de estado en el software y también en el widget de selección de articulación como se muestra en las imágenes Figura 112, 113, 114 y 115.

Figura 112

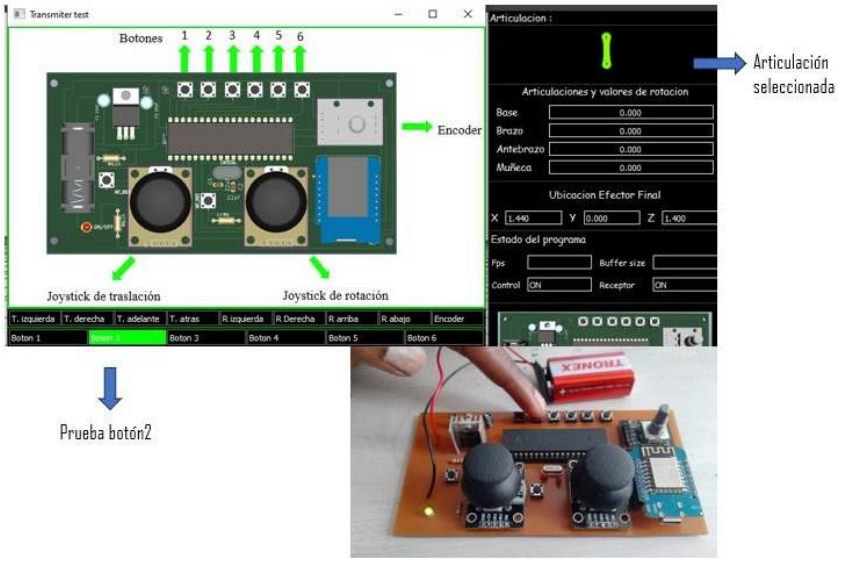
Prueba de Botón 1 y Selección de Articulación 1



Fuente. Elaboración propia.

Figura 113

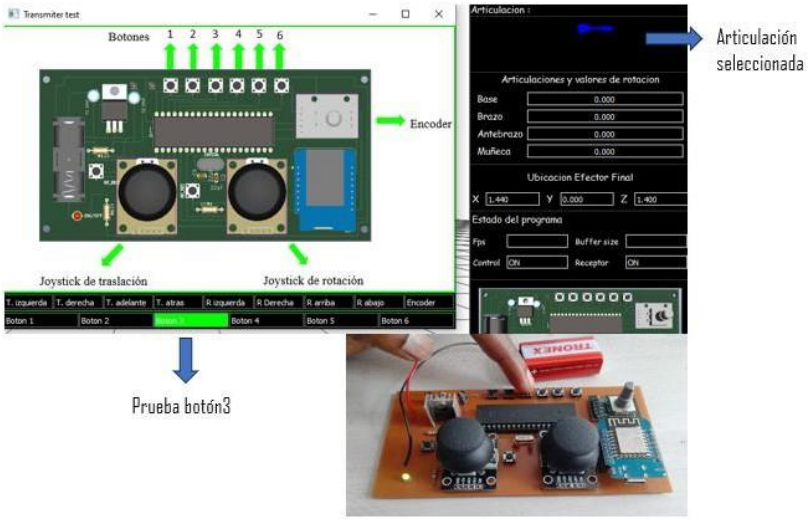
Prueba Boton2 y Selección Articulación 2



Fuente. Elaboración propia.

Figura 114

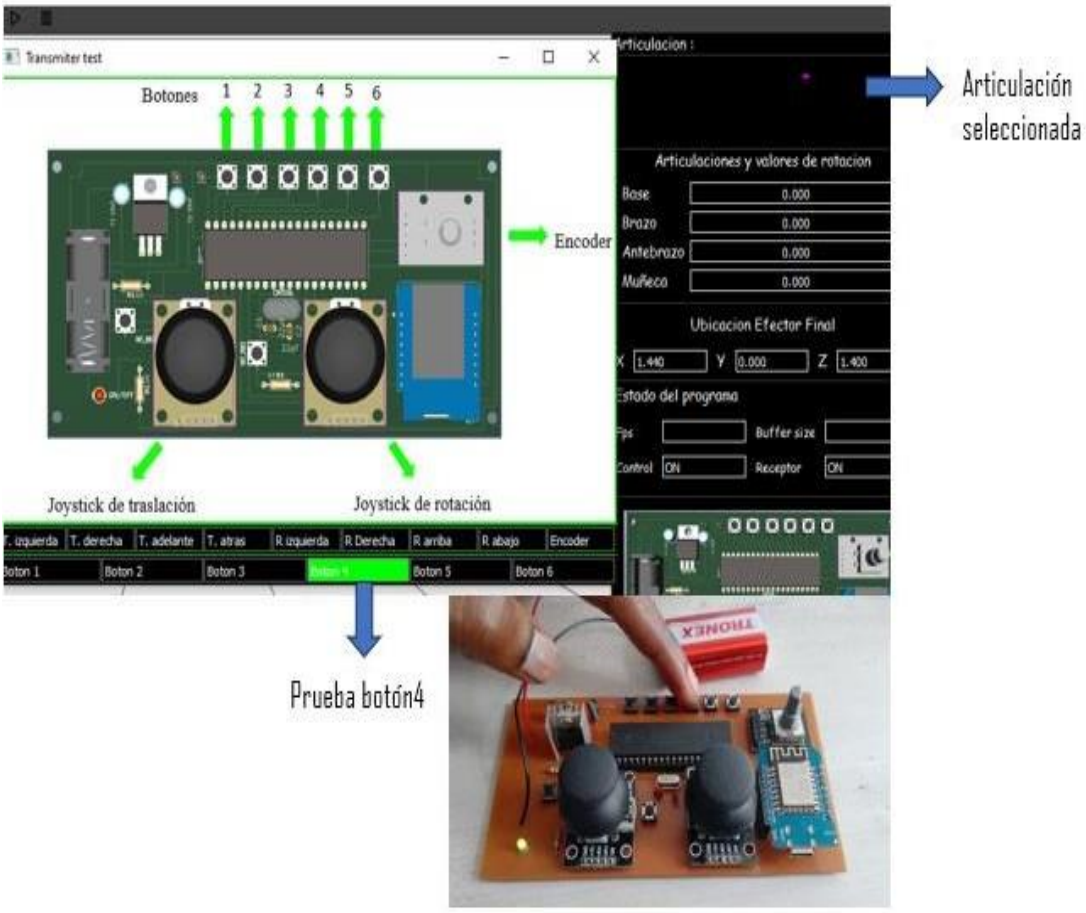
Prueba Botón 3 y Selección Articulación 3



Fuente. Elaboración propia.

Figura 115

Prueba Botón 4 y Selección de Articulación 4



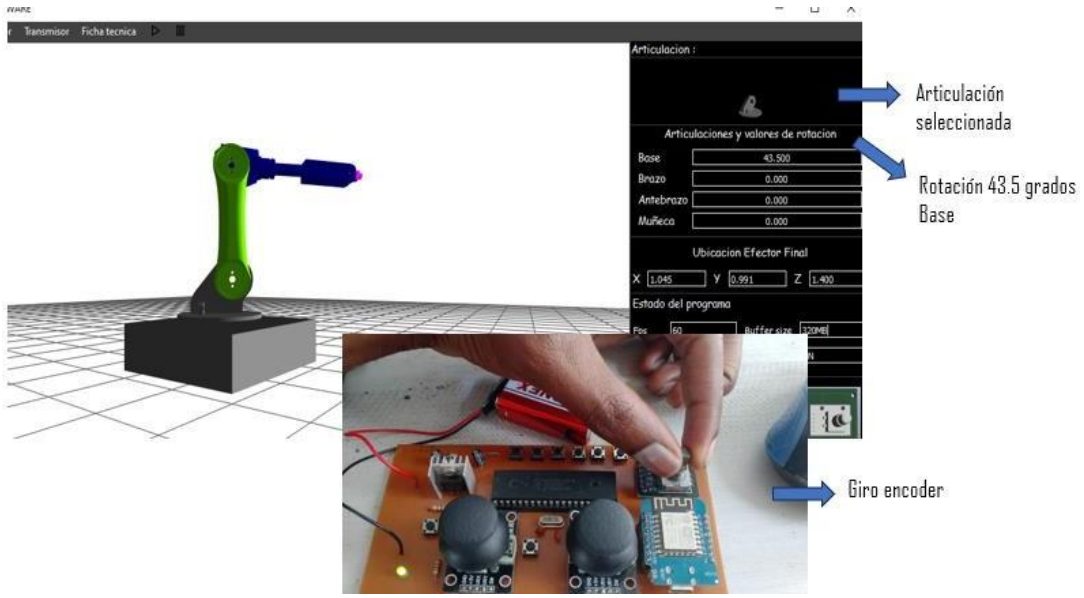
Fuente. Elaboración propia.

Prueba del Potenciómetro Digital Encoder

El potenciómetro digital encoder es el encargado de realizar las rotaciones en cada una de las articulaciones. Para realizar la prueba inicialmente se abrió la ventana de pruebas del control remoto en el software, se dejó la cámara fija en una posición, se giró el potenciómetro encoder y se analizó los cambios de rotaciones en las articulaciones en pantalla, como se puede observar en las **figuras** 116, 117, 118 y 119.

Figura 116

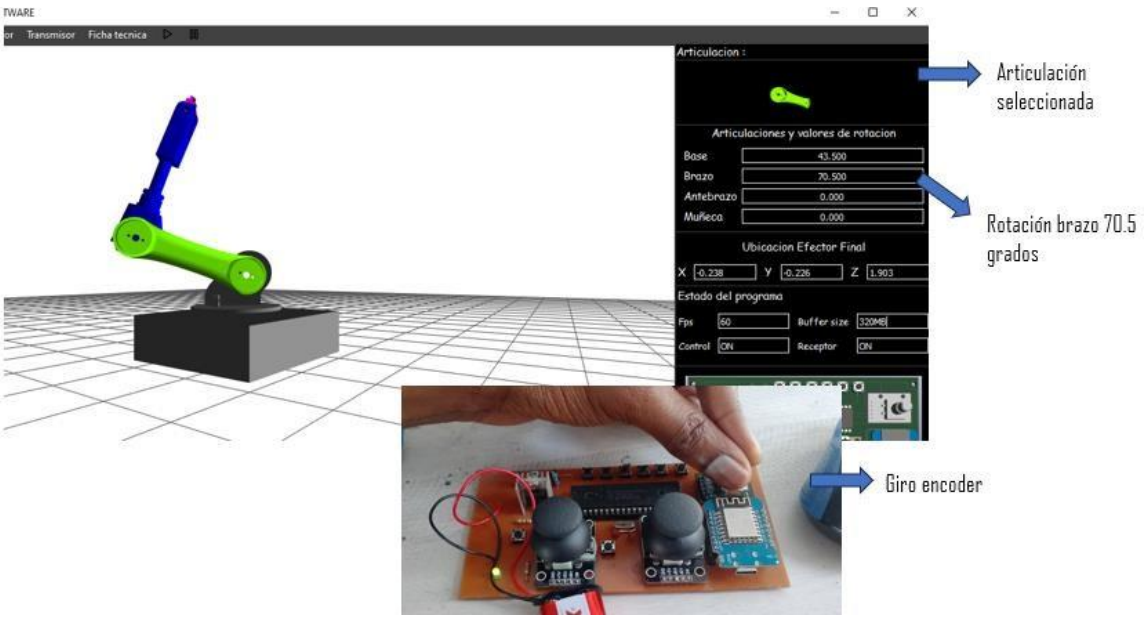
Prueba Encoder Giro Articulación Base



Fuente. Elaboración propia.

Figura 117

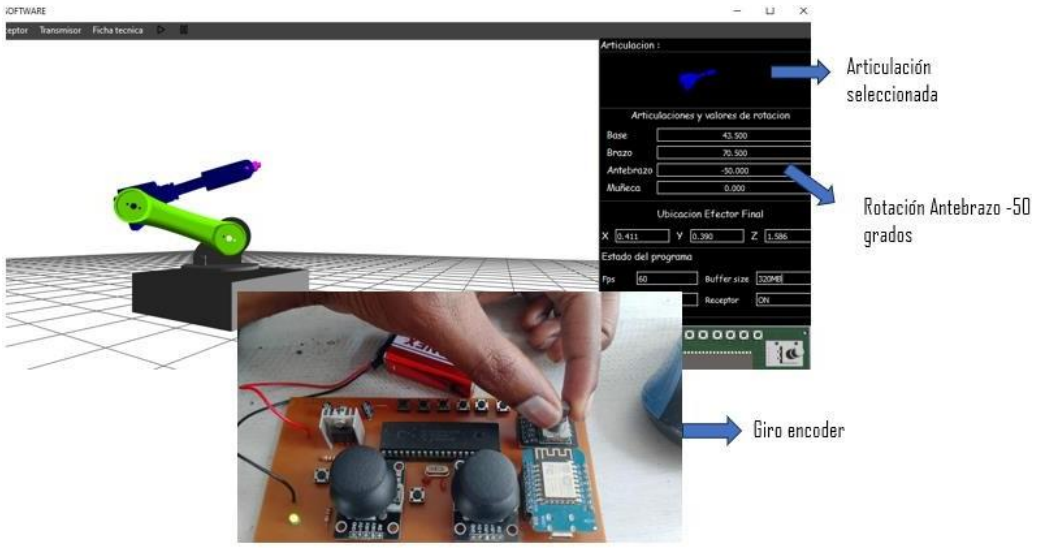
Prueba Encoder Giro Articulación Brazo



Fuente. Elaboración propia.

Figura 118

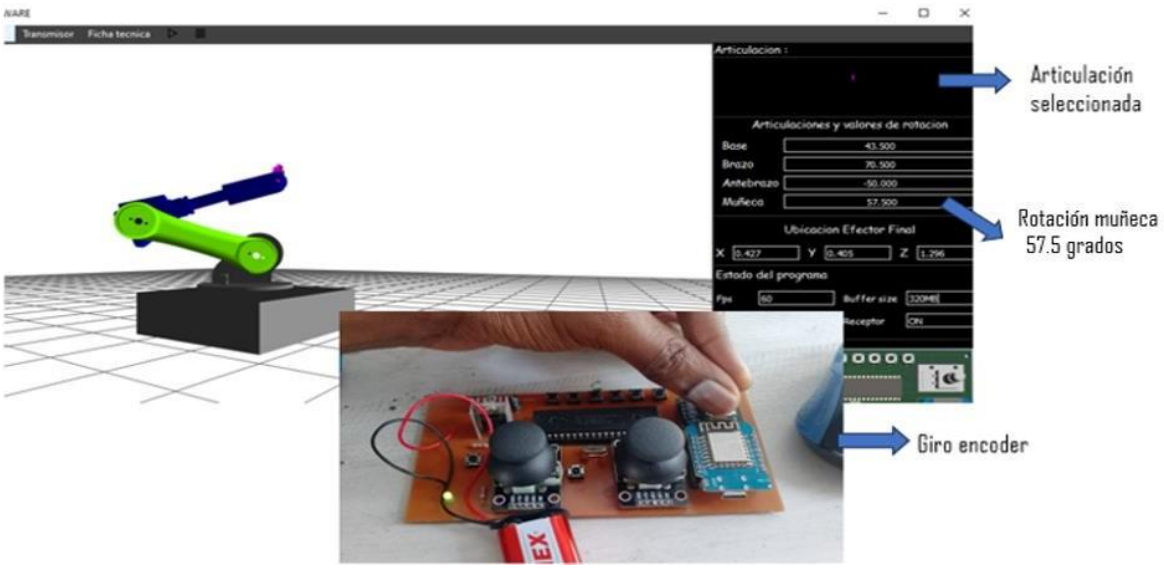
Giro Encoder Prueba Articulación Antebrazo



Fuente. Elaboración propia.

Figura 119

Prueba Encoder Articulación Muñeca



Fuente. Elaboración propia.

Recomendaciones y Futuras Mejoras

Durante el desarrollo del prototipo de software para la simulación de movimientos del brazo ROBOT NJT23 se evidenciaron futuros desarrollos que puedan potenciar la funcionalidad del sistema. Dentro de las mejoras esta:

- Sistema de batería recargable por energía solar, convirtiendo el hardware en un dispositivo amigable con el medio ambiente.
- Control remoto con componentes SMD: “Menor tamaño” haciendo que el equipo pueda ser mucho más portable.
- Cinemática inversa para los movimientos del robot NJT23 dando al usuario la posibilidad de obtener las coordenadas articulares a partir de la posición del efector final.
- Interacción de herramientas del robot con objetos del entorno, de manera que el usuario pueda interactuar con los objetos de una planta industrial.
- Diversos modelos de Robots industriales, diferentes modelos de robots con diferentes movimientos cinemáticos, eslabones y articulaciones ampliaría el aprendizaje de losusuarios
- Desarrollo de un motor grafico más realista, aprovechando las nuevas operaciones de las tarjetas gráficas se podría diseñar una escena 3D que se asemeje más a la realidad. Esto podría fortalecer la seguridad y la confianza en el aprendizaje de los nuevos usuarios.
- Consumo reducido de energía: si se tiene un menor consumo de energía las baterías darían muchas más horas de trabajo.

Los problemas encontrados durante el desarrollo se debieron a la baja demanda de algunos componentes **SMD** que se necesitan en el desarrollo del circuito Receptor. De igual manera, para el desarrollo de placas **PCB** de doble cara es necesario realizar su fabricación en

máquinas **CNC** con sus correctas herramientas. Sin embargo, es posible simular la recepción de datos con cualquier módulo de **ESP8266 y ESP32** que tenga un convertidor de interfaz Serial "**USART a USB**"

Dentro de las recomendaciones tenemos:

- No usar una batería superior a 9V, el diseño técnico del equipo soporta voltajes de entrada entre 7V a 9V.
- No modificar la velocidad de los puertos COM estáticamente en el sistema operativo: según los rangos de operación del receptor su funcionamiento esta entre los 9600 baudios a los 31250 baudios. Se recomienda dejar las configuraciones predeterminadas de los puertos COM en el sistema operativo.

Conclusiones

Gracias al software de distribución libre blender se ha logrado modelar el brazo robot virtual, teniendo un nivel básico de herramientas en el manejo de modelamiento de diseño 3D, sin embargo, para una optimización de datos en la GPU es necesario realizar un modelo bajo el concepto Low Poly o en español baja intensidad de polígonos. Esto ayudara a consumir menos recursos en el computador. Qtdesigner no es tan solo compatible con el lenguaje de programación Python. De hecho, su distribución fue primeramente lanzada para hacer aplicaciones de escritorio con el lenguaje c++.

La construcción del circuito transmisor fue un éxito sin embargo para realizar un hardware mucho más robusto se piensa utilizar algún otro dispositivo el cual soporte alguna otra interfaz de comunicación para su control como por ejemplo SPI o I2C y para él envió inalámbrico de datos algún tipo de modulación digital de corto alcance y menor costo. En previas investigaciones se analizó la posibilidad de utilizar algún otro microcontrolador de la familia microchip aún más barato, sin embargo, por temas de stock en el país solo se consiguen las referencias más comerciales de microcontroladores de 8bits.

El diseño del módulo receptor fue uno de los grandes hitos del proyecto, debido que al recibir los datos de manera inalámbrica existía la posibilidad de perder algunas variables en el medio de transmisión, en este caso el aire. Sin embargo, gracias a los algoritmos y documentos oficiales del fabricante espressif del chip ESP8266 se ha podido lograr el objetivo.

El software ha logrado su unión junto con el hardware dando por cumplido los objetivos del proyecto, es necesario contar con personal capacitado en el tema de pruebas de software para la detención de errores en caso de que se llegue a comercializar o compartir el proyecto.

Bibliografía

ABB. (01 de 02 de 2023). *ABB.* Obtenido de ABB: https://new.abb.com/products/robotics/es/robotstudio

AG, C. R. (14 de Marzo de 2024). *coppeliarobotics*. Obtenido de coppeliarobotics: https://coppeliarobotics.com/#contact

Antonio Barrientos, L. F. (2007). *Fundamentos de robotica segunda edicion.* Madrid: Mc-Graw-Hill.

Automatica, C. E. (2011). *El libro blanco de la robotica en españa.* Barcelona: CEA-GT rob2011.

Bernard Kolman, D. R. (2006). *Algebra lineal.* Ciudad De Mexico: *Pearson Educacion*.Blender. (3 de 1 de 2023). *https://www.blender.org*. Obtenido de https://www.blender.org: https://www.blender.org/about/

Brejio, E. G. (2008). *COMPILADOR C CCS Y SIMULADOR PROTEUS PARA MICROCONTROLADORES PIC.* Ciudad de Mexico: Alfaomega.

Chica, J. E. ((S, F) de (S, F) de 2020). *Universidad tecnica de ambato*. Obtenido de https://repositorio.uta.edu.ec/handle/123456789/31642

Colombia, M. T. (12 de 06 de 2021). *https://gobiernodigital.mintic.gov.co/*. Obtenido de https://gobiernodigital.mintic.gov.co/: https://gobiernodigital.mintic.gov.co/portal/Iniciativas/Software-libre/

Corella-Solís, E. (20 de 10 de 2021). *Revistas tecnologico de costa rica*. Obtenido de Revistas tecnologico de costa rica: https://revistas.tec.ac.cr/index.php/tec_marcha/article/view/5920

Cyberbotics. (28 de Junio de 2023). *Cyberbotics*. Obtenido de Cyberbotics: https://cyberbotics.com/#webots

ELECTRONICS, M. (2002). *Instruction Manual RV2-AJ.* USA: MITSUBISHIELECTRONICS.

Espressif. (16 de Octubre de 2022). *Espressif*. Obtenido de Espressif: https://www.espressif.com/

Gazebo. ((S. F) de (S. F) de (S. F)). *Gazebo*. Obtenido de https://gazebosim.org/home

Gonzalo López Nicolás, R. A. (19 de 09 de 2019). *Universidad de Zaragoza*. Obtenido de

Universidad de Zaragoza: https://deposita.unizar.es/record/50107?ln=es#

Group, K. (19 de Junio de 2022). *OpenGL*. Obtenido de OpenGL: https://www.opengl.org/ Group, L. O. (14 de 11 de 2022). *LearnOpenGL*. Obtenido de https://learnopengl.com/Getting-started/OpenGL

Honda. (23 de Enero de 2023). *Honda*. Obtenido de Honda: https://www.honda.mx/asimo

IBM. (12 de 04 de 2021). *IBM*. Obtenido de https://www.ibm.com/docs/es/aix/7.2?topic=parameters-baud-rate

Intelitek. ((S. F) de (S. F) de 2020). *Intelitek*. Obtenido de Intelitek: https://www.intelitek.com/resources/pdf/35-1005-8600_DS_HW_SCORBOT-ER4u_Ver%20K.pdf

Intelligence, M. (2024). *mordorintelligence.* Hyderabad - Telangana India: Mordor Intelligence. Obtenido de mordorintelligence: https://www.mordorintelligence.com/es/industry-reports/latin-america-warehouse-robotics-market

MathWorks. (1 de 03 de 2023). *https://la.mathworks.com/*. Obtenido de https://la.mathworks.com/: https://la.mathworks.com/company.html?s_tid=hp_ff_a_company

Microchip. (14 de 09 de 2019). *Microchip Microcontrollers*. Obtenido de Microchip Microcontrollers: https://www.microchip.com/en-us/product/PIC16F877A

MRPT. (25 de Marzo de 2024). *MRPT*. Obtenido de MRPT: https://docs.mrpt.org/reference/latest/index.html

Ortiz, P. (26 de Septiembre de 2023). *National geographic*. Obtenido de National geographic: https://historia.nationalgeographic.com.es/a/inventos-griegos-automatas-heron_9395

Qtdesigner. (22 de 01 de 2023). *Doc-Qt*. Obtenido de https://doc-qt- io.translate.goog/qtforpython-6/overviews/stylesheet- examples.html?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es-419&_x_tr_pto=sc

Roberto Hernandez Sampieri, C. F. (2010). *Metodologia de la investigacion.* Mexico Distrito Federal: McGrawHill .

ROS. (03 de 01 de 2021). *ROS*. Obtenido de ROS: https://www.ros.org/STATISTAS. (1 de Febrero de 2023). *Statistas*. Obtenido de https://es.statista.com/estadisticas/1220908/america-latina-indice-tecnologias-vanguardia-pais/

TIC, C. (29 de marzo de 2019). *INCP*. Obtenido de https://incp.org.co/uso-robots-america-latina/

UNAD, U. (15 de 2 de 2023). *Universidad UNAD*. Obtenido de Universidad UNAD: https://academia.unad.edu.co/images/escuelas/ecbti/Investigaci%C3%B3n/Grupos_por_cadena_de_formaci%C3%B3n/Cadena_de_formaci%C3%B3n_ETR.pdf

Walter, W. G. (1 de Agosto de 1951). Scientific American. *A Machine That Learns*, págs. 60-63.

Wikipedia. (13 de 3 de 2021). *https://en.wikipedia.org/*. Obtenido de https://en.wikipedia.org/: https://en.wikipedia.org/wiki/Wavefront_.obj_file

Apéndices

Apéndice A

Código Python Multiplicación Matrices

```
from sympy import *

var('cosq1 sinq1 Ex sinq1 ')

var('E1 cos90_q2 sin90_q2 E2')

var('sinq3 E3 ')

var('cosq4 sinq4 E4')


#MATRIX ARTICULACION 1

A = Matrix([[cosq1 , 0, sinq1, Ex*cosq1 ],

            [sinq1,  0,  -cosq1, Ex*sinq1],

            [0,  1,  0,  E1],

            [0,  0,  0,  1]])


#MATRIX ARTICULACION 2

B = Matrix([[cos90_q2, -sin90_q2, 0, E2*cos90_q2],

            [sin90_q2, cos90_q2, 0, E2*sin90_q2],

            [0,    0,   1, 0],

            [0, 0, 0, 1]])


C = Matrix([[sinq3, 1,  0, E3*sinq3],

            [-1,  sinq3, 0, -E3],

            [0,    0,   1, 0],
```

```
            [0, 0, 0, 1]])


D = Matrix([[cosq4, -sinq4,  0, E4*cosq4],
            [sinq4,  cosq4, 0, E4*sinq4],
            [0,    0,   1, 0],
            [0, 0, 0, 1]])
A1 = A.multiply(B)
A2 = A1.multiply(C)
A3 = A2.multiply(D)
print("Parametro 1")
print(A3[1])
#A3 = 4 X 4 MATRIZ
#N IGUAL A LOS 16 PARAMETROS DE LA MATRIZ
#print(A3[N-1])
print("Parametro Pz")
print(A3[11])
```