DISEÑO DE LAS UNIDADES DE CONTROL PARA CADA ARTICULACIÓN DEL ROBOT EDUCATIVO PROTOTIPO SIMILAR A SCORBOT ER 9 PRO BASADA EN SISTEMAS DE PROCESAMIENTO DE BAJO COSTO

LEONARDO ANDRES DÍAZ BARRETO
JEFERSON MORENO CABRERA
DIEGO MAURICIO PARRA ALMARIO

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
INGENIERÍA ELECTRÓNICA
NEIVA
2020

DISEÑO DE LAS UNIDADES DE CONTROL PARA CADA ARTICULACIÓN DEL ROBOT EDUCATIVO PROTOTIPO SIMILAR A SCORBOT ER 9 PRO BASADA EN SISTEMAS DE PROCESAMIENTO DE BAJO COSTO

DIEGO MAURICIO PARRA ALMARIO
JEFERSON MORENO CABRERA
LEONARDO ANDRES DÍAZ BARRETO

Proyecto aplicado presentado como requisito para optar por el título de ingeniero electrónico

PEDRO TORRES SILVA
DIRECTOR

UNIVERSIDAD NACIONAL ABIERTA Y A DISTANCIA
ESCUELA DE CIENCIAS BÁSICAS, TECNOLOGÍA E INGENIERÍA
INGENIERÍA ELECTRÓNICA
NEIVA
2020

Nota de Aceptación

_____

_____

_____

_____

Presidente del Jurado

_____

Jurado

_____

Jurado

Neiva 11 de diciembre del 2020

DEDICATORIA

Se hace una dedicatoria al esfuerzo de las familias, en primer lugar, porque han sido el apoyo fundamental para poder llevar a cabo todo estos logros obtenidos hasta el momento, también dar las gracias a nuestros padres, ya que en ellos se tuvo un apoyo incondicional para llegar a la culminación de este proceso, en un recorrido bastante difícil lleno de tropiezos y contratiempos, en donde ellos estuvieron siempre ahí para darnos la mano para continuar luchando y así poder cumplir todas nuestras metas, muchas gracias.

Grupo proyecto aplicado

AGRADECIMIENTOS

En primer lugar, gratitud al Padre celestial que surte todas las herramientas fundamentales para seguir luchando cada día, también dar las gracias a nuestras familias por ser la pieza primordial en la construcción de este proyecto de vida, lo cual garantizó siempre estar enfocados en la meta.
Gran agradecimiento con los tutores que a lo largo de este camino han compartido los conocimientos para ser mejores profesionales cada día.
La más sincera gratitud al tutor asesor de este proyecto, ya que sin su apoyo no habría sido posible culminar este proceso formativo.

Es claro que todos ellos fueron pieza clave para mejorar cada día, se reitera, muchas gracias a todos ellos por permitir que broten grandes profesionales, que con el transcurso de la práctica y momento tras momento lograrán superarlos para honrar sus nombres, por haber compartido tanto de todo lo bueno que pudieron expeler.

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE ILUSTRACIONES

RESUMEN

Actualmente se necesita tener nuevas proyecciones educativas para fomentar la innovación y la creatividad de los procesos académicos, es por eso por lo cual la universidad nacional abierta y a distancia fomenta la vinculación a prácticas profesionales adecuadas para que el estudiante interactúe en un medio adecuado dirigido a la solución de problemas, esto se proyecta que se diseñe un prototipo en el cual se de hardware y software necesarios para el controlador de cada articulación del robot Scorbot ER-9 Pro.

Esto se da basado en un prototipo similar para poder implementar este sistema de tal manera que se pueda basar en el robot Scorbot ER-9 Pro para aplicación de controladores, mediante creación de piezas en 3d manejo de control remoto y adecuación de sensores de posicionamiento.

Todo basado en la investigación de lo que se viene desarrollando en el proyecto de PIE_G_29_18ECBTI, que están adelantando los Ingenieros Fabian Bolívar, Jaime Rubiano y Pedro Torres, del CCAV de Neiva, tiene como objeto: “Diseño de las unidades de control para cada articulación del Robot Educativo PROTOTIPO SIMILAR a SCORBOT ER 9 Pro basada en sistemas de procesamiento de bajo costo”.

Palabras clave: Control abierto, control cerrado, cinemática directa e inversa, robots, tarjetas de control, Arduino, Raspberry, comunicación I2C, código C++, sistemas SCADA, LabVIEW.

ABSTRACT

Currently it is necessary to have new educational projections to promote innovation and creativity of academic processes, that is why the national open and distance university encourages the connection to appropriate professional practices so that the student interacts in an appropriate environment aimed at the troubleshooting, this is projected to design a prototype in which the hardware and software necessary for the controller of each joint of the Scorbot ER-9 Pro robot will be designed.

This is based on a similar prototype to be able to implement this system in such a way that it can be based on the Scorbot ER-9 Pro robot for the application of controllers, through the creation of 3d parts, remote control management and adaptation of positioning sensors.

All based on the investigation of what is being developed in the PIE_G_29_18ECBTI project, which is being carried out by the Engineers Fabian Bolívar, Jaime Rubiano and Pedro Torres, from the CCAV of Neiva, has the following objective: “Design of the control units for each joint of the Educational Robot PROTOTYPE SIMILAR to SCORBOT ER 9 Pro based on low cost processing systems”.

Keywords: Open control, closed control, forward and inverse kinematics, robots, control cards, Arduino, Raspberry, I2C communication, código C++, SCADA systems, LabVIEW.

INTRODUCCIÓN

Teniendo en cuenta que se necesitan los medios educativos adecuados para que los estudiantes de la Escuela de Ciencias Básicas, Tecnología e Ingeniería, se determinan dar solución a que se pueda poder volver a utilizar un prototipo robótico, dando posibilidades de innovación y estudio más aplicativas

Teniendo en cuenta la actualidad de la evolución de las tecnologías se puede relacionar los sistemas embebidos con el internet de las cosas de tal manera de que los sistemas embebidos son sistemas computacionales por así llamarlos en los cuales se enfoca a dar soluciones en tiempo real a muchas tareas y debido al avance se tienen que conectar a Internet para enfocarse en la relación del sistema y el avances tecnológico actual de acuerdo a que los dispositivos y los aparatos electrónicos están evolucionando a velocidades extremas dando la facilidad a las personas de acceder a circuitos integrados y componentes electrónicos en una amplia gama y a grandes distancias en tiempo real.

Es por eso por lo que la relación es tan grande para facilitar el contexto devolución entre los controladores o sistemas computacionales y la administración de las interfaces en la web, de esta manera se puede controlar las articulaciones por medio remoto.

PLANTEAMIENTO DEL PROBLEMA

Uno de los recursos con que cuenta la universidad y que puede ser útil para lograr los objetivos de calidad en su proceso de formación, es un brazo robótico de la firma INTELITEK, denominado SCORBOT ER-9 Pro, el cual puede ser usado por cualquier estudiante o docente de Ingeniería Industrial, Ingeniería Electrónica, Ingeniería de Telecomunicaciones o Tecnología en Automatización Electrónica para su práctica o laboratorio. Esto destaca la importancia de reactivar el uso de estos elementos disponibles en la UNAD, ya que es un acercamiento importante a los procesos reales de la industria. Además, en el área de control y automatización de programas como Ingeniería Electrónica y Tecnología en Automatización Electrónica, es importante conocer, configurar y manipular este tipo de equipos, pues con ello el estudiante adquiere destrezas y desarrolla habilidades en el diseño electrónico y adquiere nociones claras del control automático que puede aplicar en la industria dentro de su vida profesional.

Los controladores han presentado fallas, en el proyecto PIE_G_29_18ECBTI han propuesto la solución al tema de dicho elemento, el controlador y como parte del proyecto, está la inclusión de estudiantes y la construcción del prototipo a nivel de dispositivos electrónicos, pues el controlador es la interfaz entre usuario y máquina. Actualmente el robot no responde a los comandos que se ejecutan en el software provisto por el fabricante; dichas fallas deben ser reportadas al fabricante y/o representante en Colombia.

En concreto, para el desarrollo de robots, existen sistemas operativos dedicados, los cuales ofrecen las librerías de programación e intérpretes multilenguaje, comandos Shell, así como diversas funcionalidades de diagnóstico y *troubleshooting* los cuales puede ser embebidos dentro de la plataforma de propósito general.

Algunos de los problemas típicos asociados al desarrollo de software para el control de robots son los siguientes:

- La programación secuencial no se adapta a las necesidades de control derivadas de la interacción del robot con un mundo inherentemente asíncrono.
- Abstracción del hardware específico del robot: servos, encoders etc.
- Manejo de la complejidad en operaciones, comunicación y distribución de datos.

En el presente proyecto se recoge también como objetivo el análisis de las ventajas y desventajas en las distintas posibilidades en el desarrollo de software para el control de robots.

Finalmente se ha elaborado un modelo de robot sobre un sistema distribuido basado en hardware programable de propósito general. Se han seguido varias vertientes a la hora de llevar a cabo el diseño de este:

Programación de robot mediante la ejecución de software customizado, haciendo uso de librerías para la lectura en puerto serie y gestión de las interfaces de propósito general.
Programación de software bajo el Sistema Operativo ROS, integración mediante librerías dedicadas de plataformas de control de propósito general tales como Arduino y Raspberry Pi.
Después de una primera aproximación y estudio del estado del arte en cuanto a programación de robots, se ha optado por llevar a cabo un diseño de controlador software basado en ROS y distribuido en dos placas de propósito general:
- Placas Arduino
- Un SBC Raspberry Pi o similar

JUSTIFICACIÓN

Para el desarrollo de un prototipo de control bajo los conceptos de dispositivos de bajo costo, es necesario desarrollar de forma de cascada (Metodología *Waterfal*l) los diferentes módulos que dicho controlador posee. Para el caso del controlador del Scorbot, es necesario el diseño e implementación del módulo de control electrónico de cada una de las articulaciones, así como el protocolo de comunicaciones entre el sistema de *Gateway* que se defina y cada una de las unidades de control de cada articulación. Se cuenta con todos los avances que ya han realizado en el proyecto de investigación y se tiene acceso al brazo robótico de propiedad de la Universidad CEAD Neiva.

Es fundamental como proyecto aplicado el propiciar nuevos aportes e investigaciones en sistemas de control embebido y con unas restricciones claramente conocidas. Bajo este argumento, la propuesta se encamina a diseñar a partir de los principios cinemáticos directos como inverso, los controles para las articulaciones del brazo robótico.

La participación en la implementación de un sistema de control modular que permita a los estudiantes tener acceso a cada una de las articulaciones de forma independiente dentro de todo el sistema. En consecuencia, los estudiantes podrán no solo conocer y familiarizarse con el modelo de funcionamiento de cada articulación, sino también con los principios de funcionamiento de un brazo robótico tipo industrial.

OBJETIVOS

Objetivo general

Diseñar y desarrollar las unidades de control para cada articulación del Robot Educativo PROTOTIPO SIMILAR a SCORBOT ER 9 Pro basada en sistemas de procesamiento de bajo costo.

Objetivos específicos

Diseñar el hardware y software necesarios para el controlador de cada articulación del robot PROTOTIPO SIMILAR a Scorbot ER-9 Pro.
Diseñar e implementar la integración de los cinco (5) controles de las articulaciones a una unidad de control principal.

MARCO TEÓRICO Y CONCEPTUAL

Componentes de un Sistema Integrado

En la parte central se encuentra el microprocesador, microcontrolador, DSP, etc. Es decir, la CPU o unidad que aporta inteligencia al sistema. Según el sistema puede incluir memoria interna o externa, un micro con arquitectura específica según requisitos.

La comunicación adquiere gran importancia en los sistemas integrados. Lo normal es que el sistema pueda comunicarse mediante interfaces estándar de cable o inalámbricas. Así un SE normalmente incorporará puertos de comunicaciones del tipo RS232, RS485, SPI, I²C, CAN, USB, IP, WiFi, GSM, GPRS, DSRC, etc.

El subsistema de presentación tipo suele ser una pantalla gráfica, táctil, LCD, alfanumérico, etc.

Se denominan actuadores a los posibles elementos electrónicos que el sistema se encarga de controlar. Puede ser un motor eléctrico, un conmutador tipo relé etc. El más habitual puede ser una salida de señal PWM para control de la velocidad en motores de corriente continua.

El módulo de E/S analógicas y digitales suele emplearse para digitalizar señales analógicas procedentes de sensores, activar diodos LED, reconocer el estado abierto cerrado de un conmutador o pulsador, etc.

El módulo de reloj es el encargado de generar las diferentes señales de reloj a partir de un único oscilador principal. El tipo de oscilador es importante por varios aspectos: por la frecuencia necesaria, por la estabilidad necesaria y por el consumo de corriente requerido. El oscilador con mejores características en cuanto a estabilidad y coste son los basados en resonador de cristal de cuarzo, mientras que los que requieren menor consumo son los RC. Mediante sistemas PLL se obtienen otras frecuencias con la misma estabilidad que el oscilador patrón.

El módulo de energía (*power*) se encarga de generar las diferentes tensiones y corrientes necesarias para alimentar los diferentes circuitos del SE. Usualmente se trabaja con un rango de posibles tensiones de entrada que mediante conversores ac/dc o dc/dc se obtienen las diferentes tensiones necesarias para alimentar los diversos componentes activos del circuito.

Además de los conversores ac/dc y dc/dc, otros módulos típicos, filtros, circuitos integrados supervisores de alimentación, etc. El consumo de energía puede ser determinante en el desarrollo de algunos SE que necesariamente se alimentan con baterías y es imposible su sustitución, con lo que la vida del SE suele ser vida de las baterías.

Microprocesadores

Un microprocesador es una implementación en forma de circuito integrado (IC) de la Unidad Central de Proceso CPU de un ordenador. Frecuentemente se refiere a un microprocesador como simplemente “CPU”, y la parte de un sistema que contiene al microprocesador se denomina subsistema de CPU. Los microprocesadores varían en consumo de potencia, complejidad y coste. Los hay de unos pocos miles de transistores y con coste inferior a 2 euros (en producción masiva) hasta de más de cinco millones de transistores que cuestan más de 600 euros.

Los subsistemas de entrada/salida y memoria pueden ser combinados con un subsistema de CPU para formar un ordenador o sistema integrado completo. Estos subsistemas se interconectan mediante los buses de sistema (formados a su vez por el bus de control, el bus de direcciones y el bus de datos).

El subsistema de entrada acepta datos del exterior para ser procesados mientras que el subsistema de salida transfiere los resultados hacia el exterior. Lo más habitual es que haya varios subsistemas de entrada y varios de salida. A estos subsistemas se les reconoce habitualmente como periféricos de E/S.

El subsistema de memoria almacena las instrucciones que controlan el funcionamiento del sistema. Estas instrucciones comprenden el programa que ejecuta el sistema. La memoria también almacena varios tipos de datos: datos de entrada que aún no han sido procesados, resultados intermedios del procesado y resultados finales en espera de salida al exterior.

Es importante darse cuenta de que los subsistemas estructuran a un sistema según funcionalidades. La subdivisión física de un sistema, en términos de circuitos integrados o placas de circuito impreso (PCBs) puede y es normalmente diferente. Un solo circuito integrado (IC) puede proporcionar múltiples funciones, tales como memoria y entrada/salida.

Un microcontrolador (MCU) es un IC que incluye una CPU, memoria y circuitos de E/S. Entre los subsistemas de E/S que incluyen los microcontroladores se encuentran los temporizadores, los convertidores analógicos a digital (ADC) y digital a analógico (DAC) y los canales de comunicaciones serie. Estos subsistemas de E/S se suelen optimizar para aplicaciones específicas (por ejemplo, audio, video, procesos industriales, comunicaciones, etc.).

Hay que señalar que las líneas reales de distinción entre microprocesador, microcontrolador y microcomputador en un solo chip están difusas, y se denominan en ocasiones de manera indistinta unos y otros.

En general, un SE consiste en un sistema con microprocesador cuyo hardware y software están específicamente diseñados y optimizados para resolver un problema concreto eficientemente. Normalmente un SE interactúa continuamente con el entorno para vigilar o controlar algún proceso mediante una serie de sensores. Su hardware se diseña normalmente a nivel de chips, o de interconexión de PCBs, buscando la mínima circuitería y el menor tamaño para una aplicación particular. Otra alternativa consiste en el diseño a nivel de PCBs consistente en el ensamblado de placas con microprocesadores comerciales que responden normalmente a un estándar como el PC-104 (placas de tamaño concreto que se interconectan entre sí "apilándolas" unas sobre otras, cada una de ellas con una funcionalidad específica dentro del objetivo global que tenga el SE). Esta última solución acelera el tiempo de diseño, pero no optimiza ni el tamaño del sistema ni el número de componentes utilizados ni el coste unitario. En general, un sistema embebido simple contará con un microprocesador, memoria, unos pocos periféricos de E/S y un programa dedicado a una aplicación concreta almacenado permanentemente en la memoria. El término embebido o empotrado hace referencia al hecho de que el microcomputador está encerrado o instalado dentro de un sistema mayor y su existencia como microcomputador puede no ser aparente. Un usuario no técnico de un sistema embebido puede no ser consciente de que está usando un sistema computador. En algunos hogares las personas, que no tienen por qué ser usuarias de un ordenador personal estándar (PC), utilizan del orden de diez o más sistemas embebidos cada día.

Los microcomputadores embebidos en estos sistemas controlan electrodomésticos tales como: televisores, videos, lavadoras, alarmas, teléfonos inalámbricos, etc. Incluso un PC tiene microcomputadores embebidos en el monitor, impresora, y periféricos en general, adicionales a la CPU del propio PC. Un automóvil puede tener hasta un centenar de microprocesadores y microcontroladores que controlan cosas como la ignición, transmisión, dirección asistida, frenos antibloqueo (ABS), control de la tracción, etc.

Los sistemas embebidos se caracterizan normalmente por la necesidad de dispositivos de E/S especiales. Cuando se opta por diseñar el sistema embebido partiendo de una placa con microcomputador también es necesario comprar o diseñar placas de E/S adicionales para cumplir con los requisitos de la aplicación concreta.

Muchos sistemas embebidos son sistemas de tiempo real. Un sistema de tiempo real debe responder, dentro de un intervalo restringido de tiempo, a eventos externos mediante la ejecución de la tarea asociada con cada evento. Los sistemas de tiempo real se pueden caracterizar como blandos o duros. Si un sistema de tiempo real blando no cumple con sus restricciones de tiempo, simplemente se degrada el rendimiento del sistema, pero si el sistema es de tiempo real duro y no cumple con sus restricciones de tiempo, el sistema fallará. Este fallo puede tener posiblemente consecuencias catastróficas.

Un sistema embebido complejo puede utilizar un sistema operativo como apoyo para la ejecución de sus programas, sobre todo cuando se requiere la ejecución simultánea de los mismos. Cuando se utiliza un sistema operativo lo más probable es que se tenga que tratar de un sistema operativo en tiempo real (RTOS), que es un sistema operativo diseñado y optimizado para manejar fuertes restricciones de tiempo asociadas con eventos en aplicaciones de tiempo real. En una aplicación de tiempo real compleja la utilización de un RTOS multitarea puede simplificar el desarrollo del software.

Hoy en día existen en el mercado fabricantes que integran un microprocesador y los elementos controladores de los dispositivos fundamentales de entrada y salida en un mismo chip, pensando en las necesidades de los sistemas embebidos (bajo coste, pequeño tamaño, entradas y salidas específicas). Su capacidad de proceso suele ser inferior a los procesadores de propósito general, pero cumplen con su cometido ya que los sistemas donde se ubican no requieren tanta potencia. Los principales fabricantes son ST *Microelectronics* (familia de chips STPC), *National* (familia *Geode*), Motorola (familia *ColdFire*) e Intel.

En cuanto a los sistemas operativos necesarios para que un sistema basado en microprocesador pueda funcionar y ejecutar programas suelen ser específicos para los sistemas embebidos. Por lo tanto, encontrarse con sistemas operativos de bajos requisitos de memoria, da la ventaja de aplicaciones de tiempo real, modulares (inclusión sólo de los elementos necesarios del sistema operativo para el sistema embebido concreto), etc. Los más conocidos en la actualidad son Windows CE, QNX y VxWorks de WindRiver.

Diodos Láser

LASER es un acrónimo de Light Amplification by Stimulated Emission of Radiation. Las aplicaciones de estos diodos son muy diversas y cubren desde el corte de materiales con haces de gran energía hasta la transmisión de datos por fibra óptica.

Características: ventajas frente a los diodos LED

Los diodos láser son constructivamente diferentes a los diodos LED normales. Las características de un diodo láser son

La emisión de luz es dirigida en una sola dirección: Un diodo LED emite fotones en muchas direcciones. Un diodo láser, en cambio, consigue realizar un guiado de la luz preferencial una sola dirección.

La emisión de luz láser es monocromática: Los fotones emitidos por un láser poseen longitudes de onda muy cercanas entre sí. En cambio, en la luz emitida por diodos LED, existen fotones con mayores dispersiones en cuanto a las longitudes de onda.

Debido a estas dos propiedades, con el láser se pueden conseguir rayos de luz monocromática dirigidos en una dirección determinada. Como además también puede controlarse la potencia emitida, el láser resulta un dispositivo ideal para aquellas operaciones en las que sea necesario entregar energía con precisión.

Tomado de la página web:

http://datateca.unad.edu.co/contenidos/90022/Modulo_2013_II/index.html

Materiales descripción comercial

Los materiales utilizados para la fabricación de diodos láser son prácticamente los mismos que en diodos LED. En comunicaciones se utilizan predominantemente diodos láser que emiten en el infrarrojo. También se utilizan de luz roja.

Fotodetectores

Los componentes fotodetectores son aquellos componentes que varían algún parámetro eléctrico en función de la luz.

Todos los componentes fotodetectores están basados en el mismo principio. Si construye un componente con un material semiconductor de manera que la luz pueda incidir sobre dicho material, la luz generará pares electrón - hueco. Esta generación se realiza de manera análoga a la generación térmica de portadores. Existen tres tipos de componentes fotodetectores:

- Fotorresistencias
- Fotodiodos
- Fototransistores
- Fotorresistencias

Una fotorresistencia se compone de un material semiconductor cuya resistencia varía en función de la iluminación. La fotorresistencia reduce su valor resistivo en presencia de rayos luminosos. Es por ello por lo que también se le llama resistencias dependientes de luz (light dependent resistors), fotoconductores o células fotoconductoras.

Cuando incide la luz en el material fotoconductor se generan pares electrón - hueco. Al haber un mayor número de portadores, el valor de la resistencia disminuye. De este modo, la fotorresistencia iluminada tiene un valor de resistencia bajo.

Si se deja de iluminar, los portadores fotogenerados se recombinarán hasta volver hasta sus valores iniciales. Por lo tanto, el número de portadores disminuirá y el valor de la resistencia será mayor.

Por supuesto, el material de la fotorresistencia responderá a unas longitudes de onda determinadas. Es decir, la variación de resistencia será máxima para una longitud de onda determinada. Esta longitud de onda depende del material y el dopado, y deberá ser suministrada por el proveedor. En general, la variación de resistencia en función de la longitud de onda presenta curvas.

El material más utilizado como sensor es el CdS, aunque también puede utilizarse Silicio, GaAsP y GaP.

Arduino

Arduino es una plataforma de creación electrónica de código abierto, la cual está basada en hardware y software libre, flexible y fácil de utilizar para los creadores y desarrolladores. Esta plataforma permite crear diferentes tipos de microordenadores de una sola placa a los que la comunidad de creadores puede darles diferentes tipos de uso.

Para poder entender este concepto, primero vas a tener que entender los conceptos de hardware y software libres. El hardware libre son los dispositivos cuyas especificaciones y diagramas son de acceso público, de manera que cualquiera puede replicarlos. Esto quiere decir que Arduino ofrece las bases para que cualquier otra persona o empresa pueda crear sus propias placas, pudiendo ser diferentes entre ellas, pero igualmente funcionales al partir de la misma base. El software libre son los programas informáticos cuyo código es accesible por cualquiera para que quien quiera pueda utilizarlo y modificarlo. Arduino ofrece la plataforma Arduino IDE (Entorno de Desarrollo Integrado), que es un entorno de programación con el que cualquiera puede crear aplicaciones para las placas Arduino, de manera que se les puede dar todo tipo de utilidades.

Tomado de la página web:
http://datateca.unad.edu.co/contenidos/90022/Modulo_2013_II/index.html

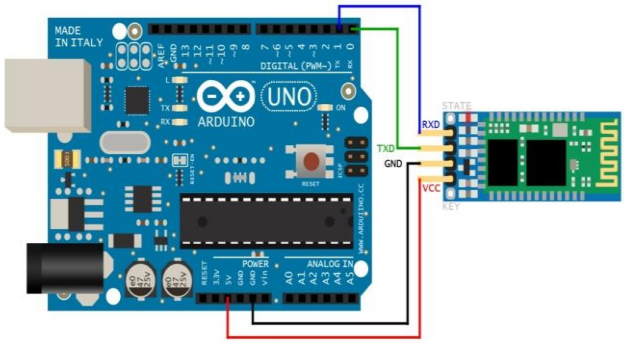


Ilustración 1 Arduino

Tomado de la página web:
http://datateca.unad.edu.co/contenidos/90022/Modulo_2013_II/index.html

Programación

La programación de Arduino es la programación de un microcontrolador. Programar Arduino consiste en traducir a líneas de código las tareas automatizadas que se desea hacer tomando datos de los sensores y en función de las condiciones del entorno programar la interacción con el mundo exterior mediante los actuadores.

Arduino proporciona un entorno de programación sencillo y potente para programar, pero además incluye las herramientas necesarias para compilar el programa y transferirlo ya compilado en la memoria *flash* del microcontrolador. Además, el IDE que ofrece un sistema de gestión de librerías y reconocimiento de placas muy práctico para la construcción del código.

Estructura de un Sketch

Un programa de Arduino se denomina sketch o proyecto y tiene la extensión ino. Importante: para que funcione el sketch, el nombre del fichero debe estar en un directorio con el mismo nombre que el sketch.

No es necesario que un sketch esté en un único fichero, pero si es imprescindible que todos los ficheros estén dentro del mismo directorio que el fichero principal.

La estructura básica de un sketch de Arduino es bastante simple y se compone de al menos dos partes. Estas dos partes son obligatorias y encierran bloques que contienen declaraciones, estamentos o instrucciones.

setup() es la parte encargada de recoger la configuración y loop() es la que contiene el programa que se ejecuta cíclicamente (de ahí el término loop –bucle-). Ambas funciones son necesarias para que el programa trabaje.

Lenguaje de Programación Arduino

El lenguaje de programación de Arduino es C++. No es un C++ puro, sino que es una adaptación que proveniente de avr-libc que provee de una librería de C de alta calidad para usar con GCC (compilador de C y C++) en los microcontroladores AVR de Atmel y muchas utilidades específicas para las MCU AVR de Atmel como avrdude.

Las herramientas necesarias para programar los microcontroladores AVR de Atmel son avr-binutils, avr-gcc y avr-libc y ya están incluidas en el IDE de Arduino, pero cuando se compila y carga un sketch se está usando estas herramientas. Aunque se hable de que hay un lenguaje propio de programación de Arduino, no es cierto, la programación se hace en C++, pero Arduino ofrece una api o core que facilitan la programación de los pines de entrada y salida y de los puertos de comunicación, así como otras librerías para operaciones específicas. El propio IDE ya incluye estas librerías de forma automática y no es necesario declararlas expresamente. Otra diferencia frente a C++ standard es la estructura del programa que se vio anteriormente.

Tomado de la página web:

http://datateca.unad.edu.co/contenidos/90022/Modulo_2013_II/index.html

ESTADO DEL ARTE

Programación:

‘La programación de Arduino es la programación de un microcontrolador. Programar Arduino consiste en traducir a líneas de código las tareas automatizadas que se desea hacer tomando datos de los sensores o valores iniciales y en función de las condiciones del entorno, programar la interacción con el mundo exterior mediante los actuadores. Arduino proporciona un entorno de programación sencillo y potente para programar, pero además incluye las herramientas necesarias para compilar el programa y transferirlo ya compilado en la memoria *flash* del microcontrolador. Además, el IDE que ofrece un sistema de gestión de librerías y placas muy práctico.’

Programación de Arduino

Este proyecto contiene Líneas de código, donde se realizan los diferentes ciclos necesarios para el movimiento del brazo, llamado de librerías tales cómo Servo, y la comunicación serial para la comunicación con Hc-05; iniciar variables, determinar número de pines de la señal de cada una de las articulaciones, cuerpo del Set Up y ciclos a cumplir en el Void Loop.

Conexión Física

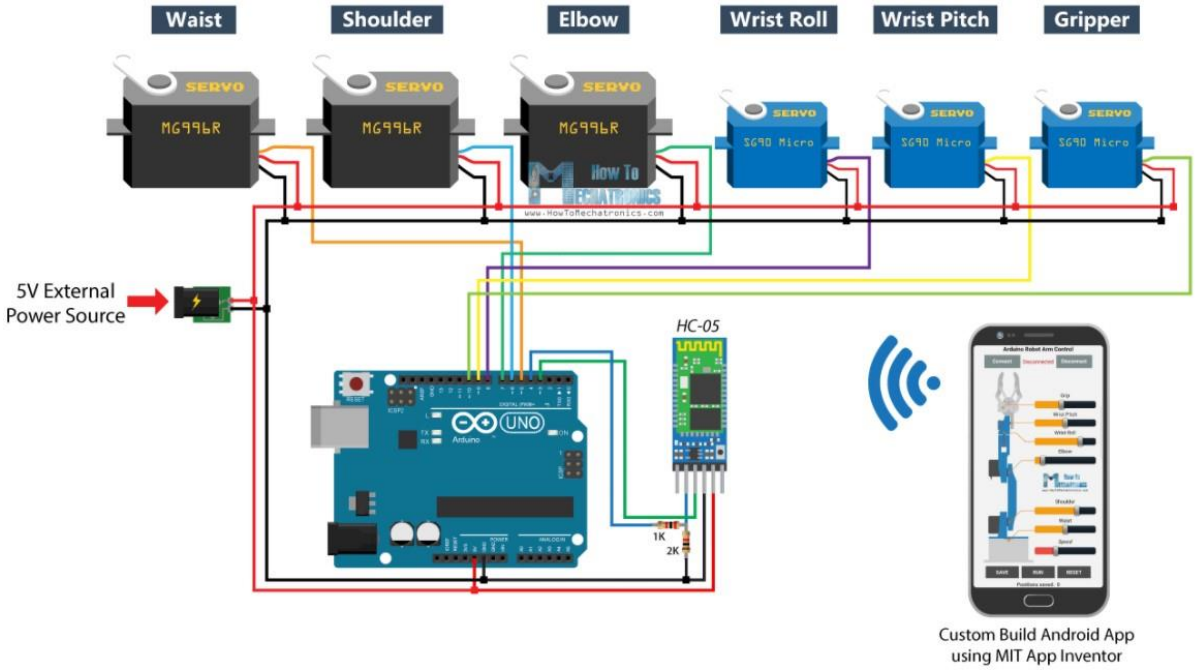


Ilustración 2 Conexión del Circuito

Para la conexión del circuito, se necesitan 6 pines digitales, estos correspondientes a las seis señales de cada uno de los servomotores (pines 5, 6, 7, 8, 9, 10); dos pines digitales para la comunicación serial tx y rx (3 y 4), los pines de 5V y GND para alimentación. La fuente externa de alimentación brinda la energía necesaria para el funcionamiento correcto de los servos, ya que el Arduino por sí solo no es capaz de sostener todos los actuadores. Estas fuentes comparten su línea negativa, debido que por aquí es donde la señal de control circula.

Teniendo en cuenta la actualidad de la evolución de las tecnologías se relaciona los sistemas embebidos con el internet de las cosas de tal manera de que los sistemas embebidos son sistemas computacionales por así llamarlos en los cuales se enfoca a dar soluciones en tiempo real a muchas tareas y debido al avance se tienen que conectar a Internet para enfocarse en la relación del sistema y el avances tecnológico actual de acuerdo a que los dispositivos y los aparatos electrónicos están evolucionando a velocidades extremas dando la facilidad a las personas de

acceder a circuitos integrados y componentes electrónicos en una amplia gama y a grandes distancias en tiempo real.

Es por lo mismo que la evolución de los sistemas inteligentes se puede decir qué parten de la sistematización que se le da a las cosas a tal punto de que se dé una asimilación humana en la cual se puede automatizar un sistema de tal manera que solucione muchos cuestionamientos habituales en donde los sistemas embebidos entran a ser parte fundamental del sistema inteligente ya que parte desde sensores controladores microprocesadores los cuales facilitan este sistema inteligente hasta llegar a tal punto de enfocarnos por así decirlo en la inteligencia artificial de ahí parte este enfoque para llegar a grandes pasos actualmente en la evolución tecnológica y relacionar los sistemas inteligentes con los sistemas embebidos. Teniendo en cuenta el campo de la ingeniería en el que se trabaja, se afirma que los sistemas embebidos han incurrido en el mundo de la robótica y de los sistemas computacionales a gran escala en donde se ha disparado una política de desarrollo e investigación cubriendo prácticamente todos los campos de producción en donde muchas veces los sistemas productivos se han disparado gracias a estos sistemas embebidos de monitoreo control y calidad adecuada para afrontar el correcto mercado actual y los avances que se han dado es por eso que estos sistemas embebidos cubren gran parte tecnológica a nivel de la ingeniería fundamentados en la robótica como principal objetivo actual de investigación para cualquier sector.

Es por eso que la relación es tan grande para facilitar el contexto de relación entre los controladores o sistemas computacionales y la administración de las interfaces en la web para que se puedan hacer correctas lecturas por ejemplo en cámaras de vídeo o comunicaciones a larga distancia tanto en tiempo real como solucionando esquema estadísticos computacionales o resolviendo problemas de por ejemplo la temperatura y la presión atmosférica de un lugar todo esto implica la conexión entre el mundo del internet y los sistemas embebidos o controladores computacionales en los cuales está enfocando el proyecto.

Módulo bluetooth HC-05

Para empezar el módulo de bluetooth HC-06 solo opera de modo esclavo, a diferencia de su hermano HC-05. Primeramente, el HC-05 ofrece una mejora con respecto a precio y características, ya que es un módulo Maestro-Esclavo, esto quiere decir, que además de recibir conexiones desde una PC o Tablet, también es capaz de generar conexiones hacia otros dispositivos bluetooth.

El módulo HC-05 es una poderosa herramienta de Bluetooth que permite ser conectando para comunicar con Arduino de manera inalámbrica y eficaz. Bluetooth tiene la enorme ventaja de estar integrado de fábrica en la mayoría de los

dispositivos. Portátiles, Tablets, y Smartphones llevan integrado Bluetooth. Además, su uso es independiente del sistema operativo (Windows, Linux, Mac o Android).

Esto convierte a la tecnología Bluetooth en uno de los mejores medios para comunicarnos de forma inalámbrica con Arduino. Por ejemplo, se puede emplear para controlar un robot desde el móvil o Tablet, o recibir mediciones de sensores en un ordenador para registrarlas en un servidor web.

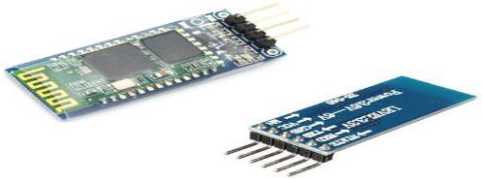


Ilustración 3 Modulo Shield Bluetooth HC-05.

La comunicación Bluetooth es similar al uso del puerto serie normal, por tanto, resulta muy versátil y sencillo de usar. La diferencia principal es que, en lugar de un conectar un cable, se empareja el módulo con el dispositivo tarjeta de desarrollo. El proceso de emparejado depende del sistema operativo (y la versión de este) pero es, en general, un proceso sencillo.
Para establecer la comunicación desde el dispositivo, se puede usar el propio Serial Monitor del Arduino IDE. También se encuentra en todos los sistemas operativos (Windows, Linux, Mac, o Android) aplicaciones para establecer la comunicación por el puerto serie.
Por último, resulta muy sencillo integrar en los programas el uso del puerto serie (y por tanto del Bluetooth), en una gran variedad de lenguajes de programación, incluidos Java, C#, VB .Net, o Python, que disponen de funciones específicas para ellos. Utilizar el módulo de Bluetooth requiere el uso de un puerto serie de nuestra placa Arduino. Por tanto, mientras se use el módulo de Bluetooth no se puede usar el puerto serie en las placas modelo Uno, Mini, y Nano. En el modelo Mega no tiene este problema, ya que incorpora 4 puertos de serie.

⚠ Mientras estemos **cargando un nuevo programa en la placa Arduino tenemos que desconectar el módulo Bluetooth**, dado que la programación se realiza a través del puerto serie.

Ilustración 4 desconexión

La conexión es sencilla:

1. Se alimenta mediante Vcc y GND.

2. Posteriormente se conecta el TXD (pin de transmisión) y RXD (pin de recepción) a los opuestos de la placa Arduino (cada TXD a un RXD).

Ilustración 5 Esquema eléctrico Módulo Shield Bluetooth HC-05.

El Arduino es una placa basada en un microcontrolador ATMEL. Los microcontroladores son circuitos integrados en los que se pueden grabar instrucciones, las cuales las escribes con el lenguaje de programación que puedes utilizar en el entorno Arduino IDE. Estas instrucciones permiten crear programas que interactúan con los circuitos de la placa.

El microcontrolador de Arduino posee lo que se llama una interfaz de entrada, que es una conexión en la que se pueden conectar en la placa diferentes tipos de periféricos. La información de estos periféricos que conectes se trasladará al microcontrolador, el cual se encargará de procesar los datos que le lleguen a través de ellos.

El tipo de periféricos que puedas utilizar para enviar datos al microcontrolador depende en gran medida de qué uso le estés pensando dar. Pueden ser cámaras para obtener imágenes, teclados para introducir datos, o diferentes tipos de sensores.

También cuenta con una interfaz de salida, que es la que se encarga de llevar la información que se ha procesado en el Arduino a otros periféricos. Estos periféricos pueden ser pantallas o altavoces en los que reproducir los datos procesados, pero también pueden ser otras placas o controladores. Además, las placas Arduino también cuentan con otro tipo de componentes llamados Escudos (Shields) o mochilas. Se trata de una especie de placas que se conectan a la placa principal para añadirle una infinidad de funciones, como GPS, relojes en tiempo real, conectividad por radio, pantallas táctiles LCD, placas de desarrollo, y un larguísimo etcétera de elementos.

Según Margolis (2011) y Jones (2017), Arduino fue creado durante 2005 por el entonces estudiante Massimo Banzi del instituto IVRAE en Italia, para integrar y facilitar el aprendizaje integrado de computación y electrónica para estudiantes de dicho instituto, con el principio de Hardware abierto (del inglés Open-Hardware).

Desde entonces, dado su fácil acceso, ya que inicialmente era gratis y luego de bajo costo, y por su gran éxito en el instituto IVRAE en Italia, se inició su distribución y comercialización tanto en Italia como en el resto del mundo. Gracias al éxito de Arduino, ya existen dispositivos hardware y aplicaciones software compatibles con Arduino. Por ejemplo, pantallas de cristal líquido y sensores de bajo costo hoy son accesibles en el mercado, así como también entornos de desarrollo integrado web como Tinkercad (2019).

La tecnología Arduino resulta en un sistema de computación con la integración directa de las áreas de hardware y software para el diseño e implementación de soluciones. Arduino hoy representa una compañía de código y hardware abierto para facilitar el acceso y uso conjunto de la electrónica y computación para el desarrollo de sistemas (Jones, 2017). Al igual que una computadora convencional, una placa Arduino puede realizar una multitud de funciones, y requiere de entradas y/o salidas para un mayor valor de su uso (Nussey, 2013). En la práctica, según (Arduino Chile, 2019), ya existen diferentes placas Arduino disponibles tales como Arduino Uno R3, Arduino Leonardo y Arduino Mega, donde los dos últimos representan una evolución desde Arduino Uno R3, en términos de mejoras e inclusión de nuevas características hardware. Por ejemplo, una versión de Arduino Mega permite la interacción con teléfonos móviles Android. A continuación, se describen detalles de componentes esenciales de Arduino Uno R3 usado en el desarrollo de los proyectos base de este trabajo: microcontrolador ATmega328; voltaje de entrada 7-12V; 14 pines digitales de I/O, con 6 salidas PWM para la generación de señales analógicas; 6 entradas análogas; 32KB de memoria Flash; Reloj de 16MHz de velocidad.

Estructura y arquitectura del robot Scorbot Er 9 pro

Debido a su estructura, el Robot SCORBOT ER 9 PRO se clasifica como angular antropomórfico, o de articulación vertical (Ilustración 1). Esta configuración es útil cuando se requiere tener acceso a espacios cerrados y para tareas de manipulación complejas. Por esta razón es la configuración más utilizada en la industria.
En la estructura del robot manipulador se distingue el brazo, formado por eslabones con sus respectivas articulaciones y el efector final, así como la base de este con el correspondiente hardware asociado. Como elementos actuadores se dispone de cinco motores de DC para el control de las 4 articulaciones y el efector final. Las estructuras de las articulaciones del Robot son todas rotacionales y con un grado de libertad (DOF), entonces, teniendo en cuenta que el robot posee cinco articulaciones, se define como un robot de cinco grados de libertad.

Ilustración 6 Brazo robótico SCORBOT ER 9 PRO. Fuente:(Intelitek, 2011).

El brazo robótico Scorbot ER 9Pro es un robot manipulador con fines educativos, diseñado y fabricado por la empresa Intelitek. Este robot tiene 5 ejes que cumplen las funciones descritas en la Tabla 2 que se lo puede apreciar esquemáticamente en la ilustración 2.

Las especificaciones técnicas del robot se presentan a continuación:

En la Tabla 1, se muestran los valores de los eslabones que conforman la estructura del manipulador, así como el rango de valores para las rotaciones de cada articulación, la apertura máxima de la pinza neumática y el peso máximo de los objetos que puede soportar.

Tabla 1 Características de los eslabones o articulaciones. Fuente:(Intelitek, 2011).

Eslabones o uniones	Longitud (mm)
1. Base y cuerpo	388
3. Brazo superior	280
4. Antebrazo	230
5. Muñeca o brida	111

6. Pinza neumática	691

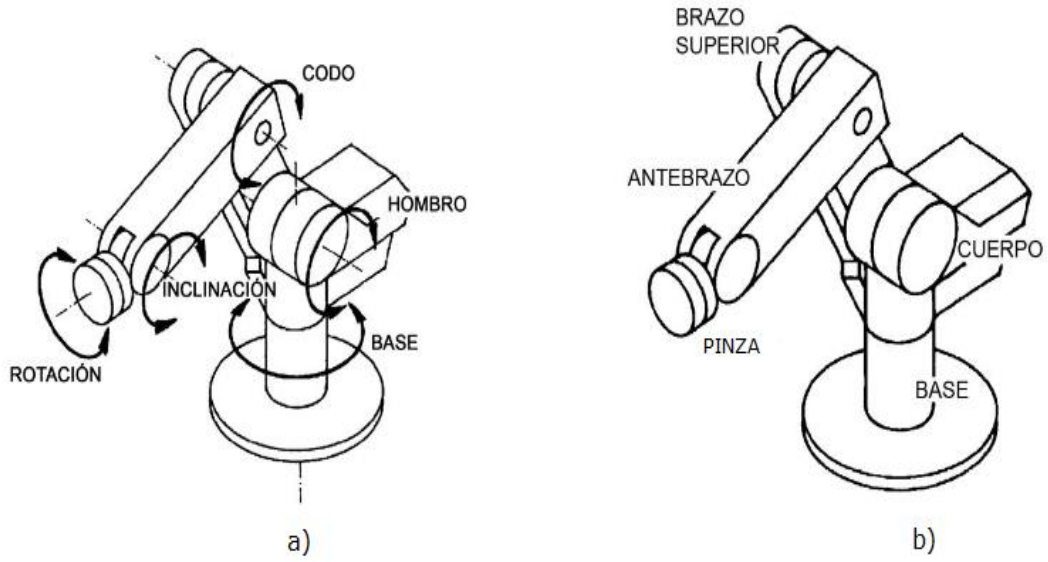


Ilustración 7 a) Articulaciones b) Eslabones o uniones del SCORBOT ER 9 PRO.
Tabla 2 Características de las articulaciones del SCORBOT ER 9 PRO

MOVIMIENTO DE EJES	RANGO	VELOCIDAD EFECTIVA	VELOCIDAD MÁXIMA
Eje1: Base	270º	80º/seg	140º/seg
Eje2: Hombro	145º	69º/seg	123º/seg
Eje 3: Codo	210º	78º/seg	140º/seg
Eje 4: Inclinación de la Muñeca o brida	196º	103º/seg	166º/seg
Eje 5: Rotación de la muñeca o brida (Efector final)	737º	185º/seg	300º/seg

Tabla 3 Movimientos de las articulaciones del robot. Fuente:(Intelitek, 2011).

Eje No.	Articulación	Movimiento	Motor No.
1	Base (Base)	Gira el cuerpo (body)	1
2	Hombro (Shoulder)	Sube y baja el brazo (upper arm)	2
3	Codo (Elbow)	Sube y baja el antebrazo (forearm)	3
4	Inclinación de la muñeca (Wrist Pitch)	Sube y baja el efector final	4
5	Rotación de la muñeca (Wrist Roll)	Gira el efector final	5

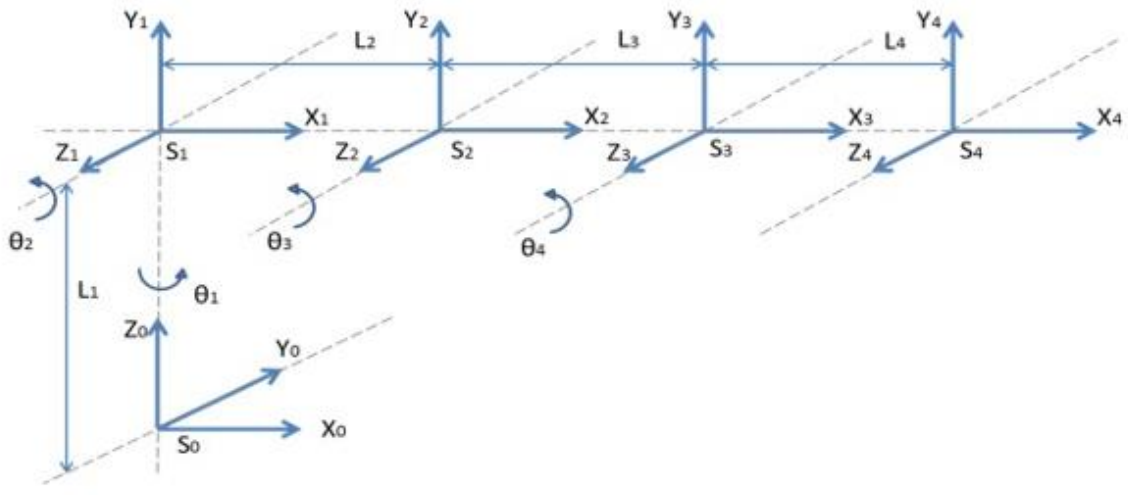


Ilustración 8 Representación del brazo robótico en sistemas de coordenadas según algoritmo de Denavit-Hartenberg.

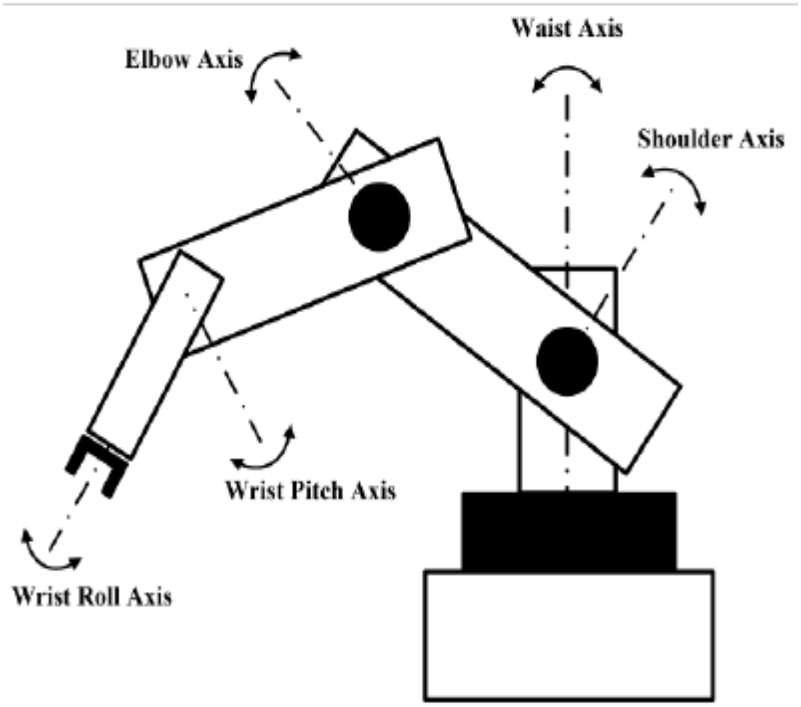


Ilustración 9 Muhammad S. A. (2014) Manipulador robótico TQ MA3000 5 DOF Fuente:(Intelitek, 2011).

Requerimientos Necesarios del Robot

Los requerimientos para un robot industrial son muy altos en términos de control de movimientos, posiciones, velocidades y aceleraciones. Los motores de un robot deben tener la capacidad de girar a diferentes velocidades y poder detenerse en cualquier instante, por lo que necesitan un amplio rango de velocidades de operación. Del mismo modo, los motores deben proveer el suficiente torque para cada velocidad del rango de funcionamiento y también para cuando el robot está detenido.

Finalmente, los motores del robot deben ser capaces de prender, detener o cambiar la dirección del movimiento de forma rápida, manteniendo la precisión en los movimientos y llegando a las posiciones finales deseadas.

Para llegar a dichos niveles de precisión, el sistema de movimiento para cada articulación tiene tres elementos principales: un servomotor eléctrico DC, una caja de reducción Harmonic Drive, y una banda con poleas.

En la ilustración 4 se presenta el sistema para cada articulación y las posiciones en las que está ubicado cada sistema en el robot. Los valores de las reducciones para cada eje.

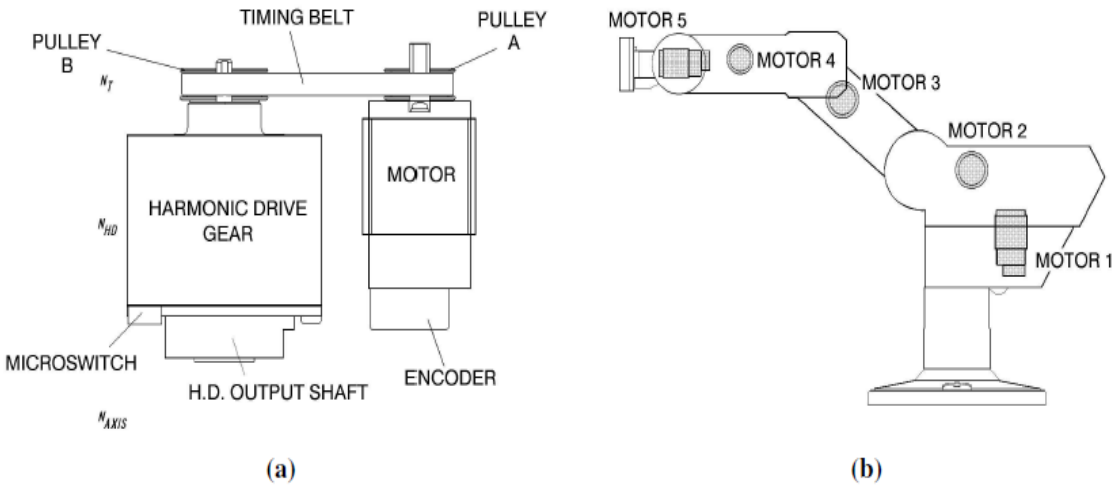


Ilustración 10 Sistema de actuadores de movimiento del robot. a) Esquema y componentes para cada articulación. b) Posición de motores para cada articulación del robot Fuente:(Intelitek, 2011).

Además, el último componente para llegar a la precisión deseada del robot son los *encoders* de cada motor. Los *encoders* son muy importantes puesto que ayudan a transmitir los pulsos eléctricos hacia el controlador y a verificar los movimientos de cada articulación. Cada *encoder* del robot trabaja con un LED, 4 fotodetectores y contiene un disco con 512 espacios para el paso de luz. Con este sistema de *encoders* se generan 2048 pulsos por cada revolución completa del eje, y por ende la resolución del *encoder* sigue la Ecuación 1 y es de 0.176°.

$$S_E = \frac{360°}{n} = \frac{360°}{512 * 4} = 0.176° \quad (1)$$

donde SE es la resolución del *encoder* y n es el número de pulsos por revolución del *encoder.*

Tabla 4 Radios de reducción mecánica en ejes del robot Fuente:(Intelitek, 2011).

Eje	R. bandas	*R. Harmonic Drive*	R. Total
1	1.33: 1	160: 1	213.33 : 1

2	1.52: 1	160: 1	243.8 : 1
3	1.33: 1	160: 1	213.33 : 1
4	1.8: 1	100: 1	180 : 1
5	-	100: 1	100 : 1

Debido a que el encoder se encuentra montado en el eje de cada motor, la resolución de cada articulación se calcula por la Ecuación 2.

$$S_{Joint} = \frac{S_E}{N_{axis}} \quad (2)$$

Donde S_{Joint} es la resolución de la articulación, S_E es la resolución del encoder y N_{axis} es la reducción mecánica del eje.
Por lo tanto, si se quiere saber la resolución para la articulación 3, basándose en la Ecuación 2 y la Tabla 4, se tendría que la resolución es de 0.000825°, y teóricamente es el mínimo posible incremento que el sistema puede identificar y controlar en ese eje [2]. Esto implica que la precisión y la confiabilidad de los movimientos del robot son altas y van a cumplir con los requerimientos de trabajo.

Principios de Control del Brazo Robótico

Mediante el volumen de trabajo del robot y los datos de distancias entre articulaciones, que se presentan en la Figura 2.3, se creó el modelo matemático para el robot a partir de los parámetros de Denavit-Hartenberg (ver 2. ESTRUCTURA Y ARQUITECTURA DEL ROBOT SCORBOT ER 9 PRO), aplicando las reglas que se presentan en Niku [3] y Spong et al. [4]. Estos parámetros se detallan en la Tabla 5, y con ellos se calculó la matriz de transformación que se presenta en la Ecuación 3.

Tabla 5 Parámetros de Denavit-Hartenberg calculados para el Scorbot ER 9Pro

T_i^j	$d[mm]$	$\theta[rad]$	$a[mm]$	$\alpha[rad]$
T_1^0	388	θ_1	70	$-\frac{\pi}{2}$
T_2^1	-43	θ_2	280	0
T_3^2	0	θ_3	230	0

$\boldsymbol{T_4^3}$	0	θ_4	0	$\frac{\pi}{2}$
$\boldsymbol{T_5^4}$	111	θ_5	0	0

Fuente:(Intelitek, 2011).

$$T_5^0 = \begin{bmatrix} -s_1 s_5 + c_1 c_5 c_{234} & -\text{c5s1} - \text{s5c1c234} & \text{c1s234} & \text{px} \\ \text{c1s5} + \text{c5s1c234} & \text{c1c5} - \text{s5s1c234} & \text{s1s234} & p_y \\ -\text{c5s234} & \text{s5s234} & \text{c234} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

Donde:

$s_i = \sin\theta_i$ para i = 1;2;3;4;5
$c_i = \cos\theta_i$ para i = 1;2;3;4;5
$s_{23} = \sin(\theta_2+\theta_3)$
$c_{23} = \cos(\theta_2+\theta_3)$
$s_{234} = \sin(\theta_2+\theta_3+\theta_4)$
$c_{234} = \cos(\theta_2+\theta_3+\theta_4)$
$p_x = a_1c_1-d_2s_1+a_2c_1c_2+d_5c_1s_{234}+a_3c_1c_{23}$
$p_y = a_1s_1+d_2c_1+a_2s_1c_2+d_5s_1s_{234}+a_3s_1c_{23}$
$p_z = d_1-a_2s_2+d_5c_{234}-a_3s_{23}$

A partir de estos conceptos básicos se realizarán las consideraciones técnicas, criterios de diseño, para alcanzar el objetivo de este trabajo.
Del documento de trabajo del proyecto PIE_G_29_18ECBTI [5]: "*Diseño y desarrollo de una unidad de control para el Robot Educativo SCORBOT ER 9 Pro basada en sistemas de procesamiento de bajo costo".* Se han tomado los siguientes elementos para el desarrollo teórico:

A. ESPECIFICACIONES DEL SISTEMA

1. Especificaciones generales del brazo robótico SCORBOT ER 9 PRO (Tabla No. 6)
2. Por los requerimientos y limitaciones del trabajo, el robot funciona con dos tipos de motores dependiendo de cada uno de los ejes, y sus características se presentan en la tabla No. 7.
3. Sistema de Reducción *Harmonic Drive*

Este sistema de reducción tiene cuatro componentes importantes (Ilustración 6 a)):

- **Anillo circular**: es un aro sólido de acero dentado interiormente, y sujeto al eslabón en el que se requiere el movimiento final de un determinado motor.

- **Generador de onda**: es un disco sólido elíptico, que se mueve directamente con el eje del motor.
- **Anillo flexible**: es un cilindro flexible muy delgado, dentado exteriormente. Transmite el movimiento mediante sus dientes.
- **Anillo dinámico**: es un cilindro sólido de acero con dientes internos.

Tabla 6 Especificaciones del brazo robot y los motores del robot Fuente:(Intelitek, 2011).

Especificaciones del brazo robot	
Estructura mecánica	Articulación vertical, protección hermética
Número de ejes	5 más la pinza
Rango del eje	• Eje 1: Rotación de la base: 270° • Eje 2: Rotación del hombro: 145° • Eje 3: Rotación del codo: 210° • Eje 4: Inclinación de la muñeca: 196° • Eje 5: Rotación de la muñeca: 737°
Velocidad: Velocidad efectiva, Velocidad Máxima	• Eje 1: Rotación de la base — 80°/seg, 140°/seg • Eje 2: Rotación del hombro — 69°/seg, 123°/seg • Eje 3: Rotación del codo — 77°/seg, 140°/seg • Eje 4: Inclinación de la muñeca — 103°/seg, 166°/seg • Eje 5: Rotación de la muñeca — 175°/seg, 300°/seg
Radio máximo de operación	691 mm (27.2") sin pinza
Efector final: opciones:	Pinza neumática
	Servopinza eléctrica de CC
Homing (Retorno a la posición de inicio)	Interruptor óptico y codificador con pulso de referencia en todos los ejes
Retroalimentación	Codificadores ópticos incrementales con pulso de referencia en todos los ejes
Actuadores	Servo motor de 24VCC en todos los ejes
Transmisión	Engranajes armónicos y correas dentadas
Carga máxima	•5 kg (11 lbs.) (con aceleración reducida) •2 kg (4,4 lbs.) (Velocidad máxima) (incluida la pinza)
Repetibilidad de la posición	±0,05 mm (0.002")
Peso	53,5 kg (117,7 lbs.)
Temperatura ambiente de operación:	2° - 40°C (36° - 104°F)

Entradas digitales	**16**
Salidas digitales	16

Salidas analógicas	2

	Ejes de motor 1, 2, 3	Ejes de motor 4, 5
Par máximo	143 onzas /pulgadas	27,8 onzas /pulgadas
Par nominal	32 onzas /pulgadas	12,5 onzas /pulgadas
Velocidad máxima de operación	4000rpm	4500rpm
Peso	1,29kg / 2,84lbs	0,28kg / 0,62lbs

Fuente:(Intelitek, 2011).

Cuando el generador de onda gira, por su forma elíptica, deforma al anillo flexible y hace que con sus dientes toque en dos puntos del anillo circular, éste a su vez realiza un giro mínimo en comparación con el giro del generador de onda. En la Ilustración 6, b) se muestra un ejemplo de cómo funciona el sistema y como se reduce la velocidad de giro. En el ejemplo, se distingue claramente que después de una rotación completa del generador de onda, el anillo circular solo se movió la rotación equivalente a dos dientes.

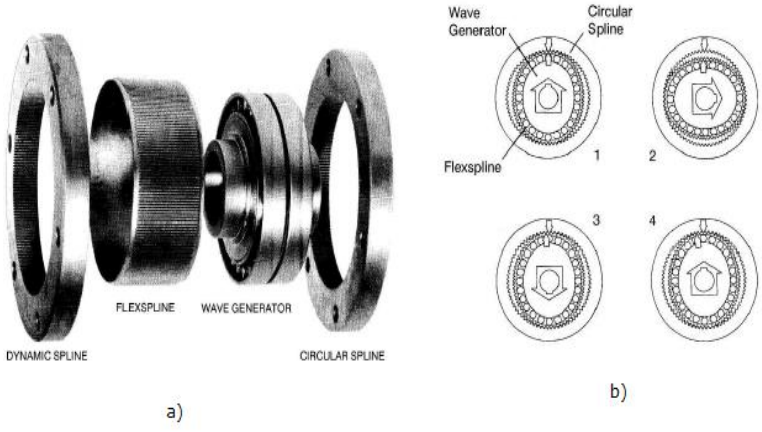


Ilustración 11 Sistema de reducción mecánica Harmonic Drive. a) Componentes del sistema. b) Funcionamiento del sistema. Fuente:(Intelitek, 2011).

4. **Encoders**

Los encoders usados en el Scorbot ER 9Pro contienen un solo LED como fuente de luz. Opuesto al LED se ubica un circuito integrado de detección de luz (Ilustración 7b), que contiene varios fotodetectores con circuitos que producen una señal digital. Mientras el disco del *encoder* (Ilustración a) gira, el haz de luz del LED se interrumpe y no llega a los fotodetectores.

Esto da como resultado una serie de pulsos recibidos, que después son alimentados mediante un circuito de procesamiento de señal y generan las señales A, A, B, B, I y I.

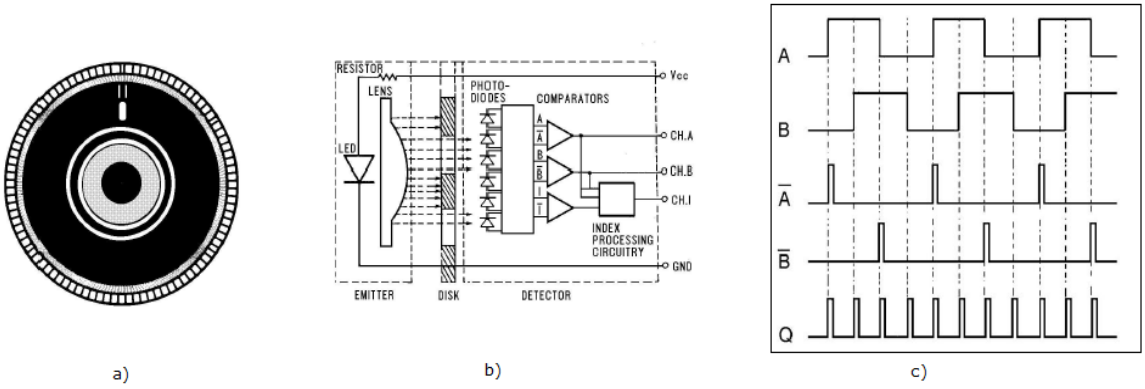


Ilustración 12 Esquematización del sistema de encoders. a) Disco perforado del encoder. b) Circuito del encoder. c) Señales de salida del encoder. Fuente:(Intelitek, 2011).

Finalmente, los comparadores reciben las señales y producen tres salidas digitales para los canales A, B e I. Los canales A y B tienen un desfase de 90 grados que indica el sentido de giro: si A llega primero que B, el sentido es antihorario, e inversamente, el sentido de giro es horario si B lidera a A.

5. Tarjetas SBC

Tabla 7 2019 Best Single Board Computers (Raspberry Pi Alternatives). Valor en dolares:

SBC	Processor	GPU	Memory	Market Price
Raspberry Pi Zero W	Broadcom BCM2835 (1x ARM1176JZFS core @ 1GHz)	VideoCore IV dual-core	512MB SDRAM	$ 10
Onion Omega2Plus	580 MHz MIPS	n/a	128 MB	$ 13

Rock64 Media Board	Rockchip RK3328 (4x Cortex-A53 @ 1.5GHz)	Mali-450 MP2	1GB, 2GB, or 4GB DDR3L; empty eMMC slot	$ 25
PocketBeagle	Octavo Systems OSD335x SiP with TI Sitara AM3358 (1x Cortex-A8 @ 1GHz)	PowerVR SGX530	512MB RAM	$ 25
Arduino Mega 2560	ATmega2560	n/a	256KB Flash ROM	$ 33
Le Potato	Amlogic S905X (4x Cortex-A53 @ up to 2GHz)	Mali-450 MP2	1GB or 2GB DDR3 RAM; optional 8GB to 64GB eMMC	$ 35
BBC micro:bit	ARM Cortex-M0	n/a	256KB Flash ROM, 16KB RAM	$ 37
Pine A64-LTS	Allwinner R18 (4x Cortex-A53 cores @ 1.2GHz)	Mali-400 MP2	2GB DDR3 RAM; optional up to 128GB eMMC	$ 50
Banana Pi M64	Allwinner A64 (4x Cortex-A53 @ 1.2GHz)	Mali-400 MP2	2GB DDR3 RAM; 8GB to 64GB eMMC	$ 61
Odroid-C2	Amlogic S905 (4x Cortex-53 @ up to 1.5GHz)	Mali-450 MP2	2GB DDR3 RAM; optional 8GB eMMC	$ 62
Orange Pi Plus2	Allwinner H3 (4x Cortex-A7 @ 1.6GHz)	Mali-400 MP2	2GB DDR3 RAM; 8GB eMMC	$ 67
Rock Pi 4 Model B	Rockchip RK3399	Mali T860MP4	64bit dual channel LPDDR4@3200Mb/s, 1GB/2GB/4GB optional	$ 70
NanoPC-T3 Plus	Samsung S5P6818 (8x Cortex-A53 @ 400MHz to 1.4GHz)	Mali-400 MP	2GB DDR3 RAM; 16GB eMMC	$ 75

Odroid-XU4	Samsung Exynos5422 (4x Cortex-A15 @ 2.0GHz and 4x Cortex-A7 @ 1.4GHz)	Mali-T628 MP6	2GB LPDDR3 RAM; optional up to 64GB eMMC	$ 85
Asus Tinker Board S	Rockchip RK3288 (4x Cortex-A17 @ 1.8GHz)	Mali-T760	2GB LPDDR3 RAM	$ 89
LattePanda	Intel Cherry Trail Z8350 Quad Core 1.8GHz	Intel HD Graphics @200-500 Mhz	2GB DDR3L	$ 119
Minnowboard Turbot Dual Ethernet	Intel Atom E38xx Series	Gen 7 (4 Execution Units)	2GB DDR3L 1067MT/s, on board	$ 170
BeagleBoard-X15	TI AM5728 2×1.5-GHz ARM Cortex-A15	Dual Core SGX544 , 532 MHz	2GB DDR3L RAM	$ 240
Huawei HiKey 960	Kirin 960 (4 x 2.3GHz ARM A73 cores, and 4 x 1.8GHz ARM A53 cores)	ARM Mali G71 MP8	3GB LPDDR4 SDRAM	$ 299

Fuente: https://all3dp.com/1/single-board-computer-raspberry-pi-alternative/

Por las variadas posibilidades de conseguirse en el mercado la placa de desarrollo, se debe tomar una decisión al respecto, dentro de los tiempos establecido y de acuerdo con la viabilidad que ofrezca.

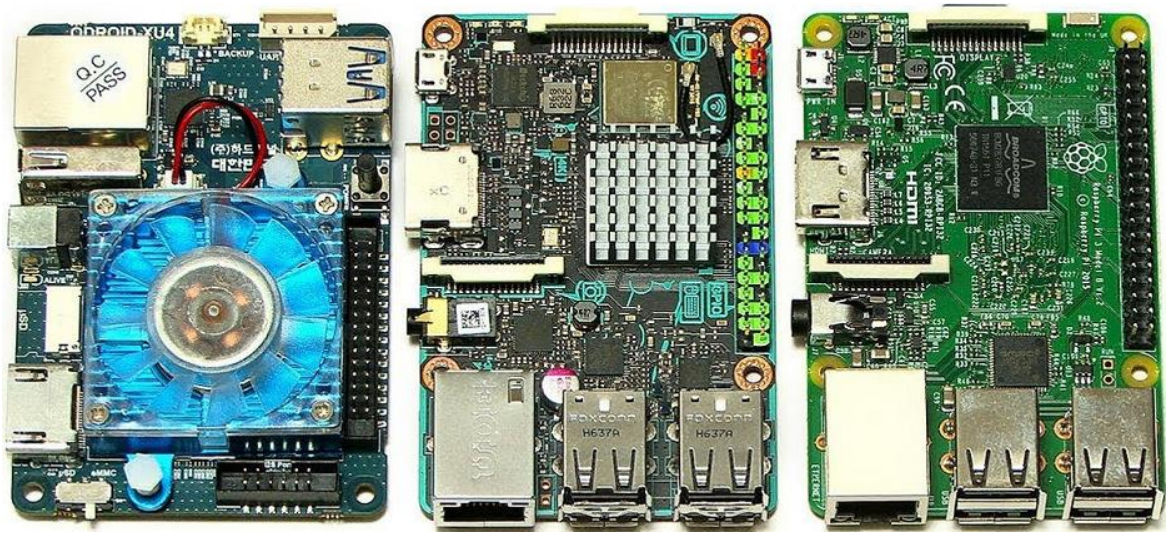


Ilustración 13 Ejemplo de tarjetas SBC más populares

Specifications

- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 1GB, 2GB or 4GB LPDDR4-3200 SDRAM (depending on model)
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 USB 3.0 ports; 2 USB 2.0 ports.
- Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards)
- 2 × micro-HDMI ports (up to 4kp60 supported)
- 2-lane MIPI DSI display port
- 2-lane MIPI CSI camera port
- 4-pole stereo audio and composite video port
- H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)
- OpenGL ES 3.0 graphics
- Micro-SD card slot for loading operating system and data storage
- 5V DC via USB-C connector (minimum 3A*)
- 5V DC via GPIO header (minimum 3A*)
- Power over Ethernet (PoE) enabled (requires separate PoE HAT)
- Operating temperature: 0 – 50 degrees C ambient

* A good quality 2.5A power supply can be used if downstream USB peripherals consume less than 500mA in total.

Ilustración 14 Características de la tarjeta RASPBERRY PI 4

Fuente: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/

DISEÑO METODOLÓGICO

La metodología se despliega a partir de una exploración al manejo preventivo y correctivo del brazo robótico mediante un prototipo de ayuda, en la cual se maneja de manera longitudinal con orientación mixta. Se fomenta el tipo de indagación de

campo porque se tomaron antecedentes con diferentes tecnologías directamente en el origen de estudio, generalmente acerca de las características, variables o secuencias que no se pueden cimentar subjetivamente, además se afirma en datos preliminares para poder planificar el proyecto a realizar. Se torna longitudinal por que las variables se evalúan en diferentes etapas o periodos, se selecciona la indagación de tal manera que esto quede como una proyección hacia otros estudiantes y su aplicación, su orientación mixta se da porque se fomentan variables cualitativas y cuantitativas. Para realizar la investigación se puede inferir que lo mencionado anteriormente parte particularmente de cuatro etapas: planificación, diseño, ejecución y evaluación.

De acuerdo con lo propuesto en el proyecto original: “El desarrollo del proceso investigativo se realizará por fases, las cuales podrán ser interactivas, siguiendo el presupuesto del modelo SCRUM. El desarrollo incremental de los requisitos del proyecto en bloques temporales cortos. Se prioriza, el concepto de bajo costo y a su vez, eficiente, realizando un control empírico del proyecto. A través del resultado real obtenido. Adaptándose a los criterios de diseño propuestas en el marco teórico. Que, a su vez, potencializa los resultados esperados. Para luego, sistematizar toda la información, y generar los siguientes productos.”

Se ha definido los siguientes bloques:

1. Diseño del control para cada articulación
    a. Diseño del hardware y software necesarios para el funcionamiento
    b. Prueba en el brazo robótico
    c. Integración por control remoto mediante Diseño de aplicación App para controlarlo desde un smartphone
    d. Fabricación de las piezas mediante impresión en 3D

2. Integración de los cinco (5) controles de las articulaciones a una sola unidad de control.
    a. Interconexión entre las articulaciones
    b. Conexión al bloque de control principal, al Raspberry Pi 3 B+ mediante protocolo I2C

RECURSOS NECESARIOS

Tabla 8 Recursos necesarios

RECURSO	DESCRIPCIÓN	PRESUPUESTO
1. Equipo Humano	Estudiantes UNAD	N. A.
2. Equipos y Software	Robot SCORBOT ER-9 Pro	N. A.
3. Viajes y Salidas de Campo	Visitas a industrias e instituciones	150000
4. Materiales y suministros	3 Servo motor mc	73500
	3 Servo motor sgr	26500
	Fuente 5v 3ª	18000
	3 Jumper 30 cm MH	9000
	2 Jumper 10 cm MH	3000
	1 Jumper 20 cm MH	2000
	bluetooth	18000
	Arduino	25000
	protoboard	12000
	7 de 3x20 mm tornillos con 9 tuercas para la pinza	5000
	12 tornillos golosos cortos para sujetar el brazo	2000
	tabla base	8000
5. Fletes	Transporte de los materiales	8700
6. Bibliografía	Búsqueda en Internet y Bases de datos	N. A.
TOTAL	360.700	

Fuente: propia

CRONOGRAMA

Tabla 9 Cronograma

ACTIVIDAD	MES 1	MES 2	MES 3	MES 4	MES 5	MES 6
Revisión bibliográfica de los investigadores principales	X					
Definición de los criterios de diseño	X	X				
Diseño e implementación del prototipo de una articulación		X	X			
Pruebas de las articulaciones			X			
Integración de las articulaciones y el bloque de control principal				X		
Pruebas finales					X	
Documentación final						X

Fuente: propia

RESULTADOS ESPERADOS

Tabla 10 Resultados esperados

RESULTADO/PRODUCTO ESPERADO	INDICADOR	BENEFICIARIO	
Fabricación de las piezas mediante impresión en 3D	Funcionalidad con respecto a lo requerido	UNAD/ académica	Comunidad
Adecuación de Sensores para el posicionamiento de cada articulación	Manejo de articulaciones de acuerdo con la programación	UNAD/ académica	Comunidad
Diseño de aplicación App para controlarlo desde un smartphone	Manejo independiente mediante dispositivo evaluando la versatilidad de los sistemas embebidos	UNAD/ académica	Comunidad
Unidad de control para cada una de las articulaciones del brazo	5 controladores	UNAD/ académica	Comunidad

robótico SCORBOT ER 9 PRO		
Software para la unidad de control de cada articulación	5 programas de control	UNAD/ Comunidad académica

Fuente: propia

DISEÑO Y CONSTRUCCIÓN

Materiales

Siendo este proyecto, un proyecto de similitud al SCORBOT, se establecieron los materiales relacionados a continuación:

Impresora 3D y Filamento Pla

Ilustración 15 Impresora y filamento PLA utilizado para las piezas

El filamento PLA, ácido poliláctico, es un termoplástico fabricado a base de recursos renovables como el almidón de maíz, raíces de tapioca o caña de azúcar. Este material considerado polímero semicristalino tiene una temperatura de fusión de 180 ºC. PlA. Este material económico, permite formarlo por medio de una impresora 3D de acuerdo con la necesidad. Teniendo un valor alto en su módulo de elasticidad (Modulo de Young) de 3.5GPa características mecánicas PLA, el que permite crear piezas lo suficientemente resistentes para el brazo, a una economía asequible.

El proyecto consta de diferentes piezas que se muestran en el ítem "Diseño", pero que a continuación se relaciona el costo, tiempo de impresión, energía utilizada, cantidad de material.

COSTO DE IMPRESIÓN 3D		
Tiempo de impresión	35	Horas
Peso Material	280	Gramos
kw/h (Electrohuila)	620,57	Pesos
Potencia de impresión	0,27	kw/h
Costo Material	$ 23.800,00	Pesos
Costo Energia	$ 5.864,39	Pesos
Seguro	$ 5.933	Pesos
Total	$ 71.195	Pesos
	$ 70.000	Pesos

Ilustración 16 Características de la Impresión

Teniendo un tiempo de impresión de 35 horas, se obtienen las diferentes piezas con un peso total de 280 gramos, donde las piezas generadas deben pasar por un proceso de enfriamiento y alistamiento para poder empezar a utilizarlas.

ServoMotor MG996R

Ilustración 17 Servo Motor MG996R

Se presentan sus características:

- El voltaje de funcionamiento es típicamente + 5V
  Corriente: 2.5A (6V)
- Par de bloqueo: 9,4 kg / cm (a 4,8 V)
  Par máximo de bloqueo: 11 kg / cm (6 V)
- La velocidad de funcionamiento es de 0,17 s / 60 °
  Tipo de engranaje: Metal
- Rotación: 0 ° -180 °
  Peso del motor: 55 g
- El paquete incluye cuernos y tornillos de engranajes

- *Datasheet* Servo
  Valores ideales y por encima de los necesarios para el brazo, haciendo el
- dispositivo seguro y resistente para sus tareas. Su costo está sobre los 24.000 mil pesos.
-

- Servomotor Micro SG 90

Ilustración 18 Micro Servo SG 90

Se presentan sus características:

- Dimensiones (L x W xH) = 22.0 x 11.5 x 27 mm (0.86 x 0.45 x 1.0 pulgadas)
- Peso: 9 gramos
- Peso con cable y conector: 10.6 gramos
- Torque a 4.8 volts: 16.7 oz/in o 1.2 kg/cm
- Voltaje de operación: 4.0 a 7.2 volts
- Velocidad de giro a 4.8 volts: 0.12 seg / 60 º
- Valores menores al servo MG996R, pero suficientes para diferentes articulaciones del brazo.

Materiales Eléctricos

Ilustración 19 Jumpers de Conexión

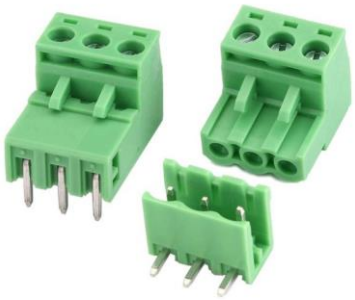


Ilustración 20 Broneras del circuito

Estos elementos importantes para la unión de los diferentes dispositivos, los jumpers para realizar los diferentes puentes y las borneras para la conexión de los cables en la protoboard y Arduino.

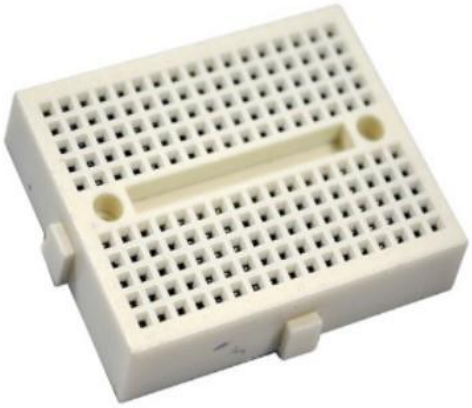


Ilustración 21 Protoboard

Elemento importante para la conexión de los dispositivos, esta tabla de conexión presta ayuda con sus líneas de continuidad, pines de fácil contacto con los jumpers.

Ilustración 22 Resistencia 2K

Materiales e insumos:
Elementos esenciales la hora de la construcción de los circuitos.

- Base en Madera
- Cauchos
- Amarraderas

Materiales de programación y conexión nombrados anteriormente en el capítulo uno, se cuenta con:

- Arduino UNO
- Modulo Bluetooth Hc-05
- Dispositivo Inteligente
- Cable de datos Arduino

Diseño

Para el diseño del brazo, se utiliza el software SOLIDWORKS, plataforma donde se llevaron los diseños borrador a los planos del dispositivo. Aquí se creó pieza por pieza del brazo, teniendo en cuenta el sistema MSG (metro, segundo, gramo). Diez piezas cuidadosamente elaboradas, con medidas exactas crean el ensamblaje de

la figura número 2.9, se simularon también los servos para crear los agujeros de los tornillos y para que éste mismo encaje de la mejor manera.

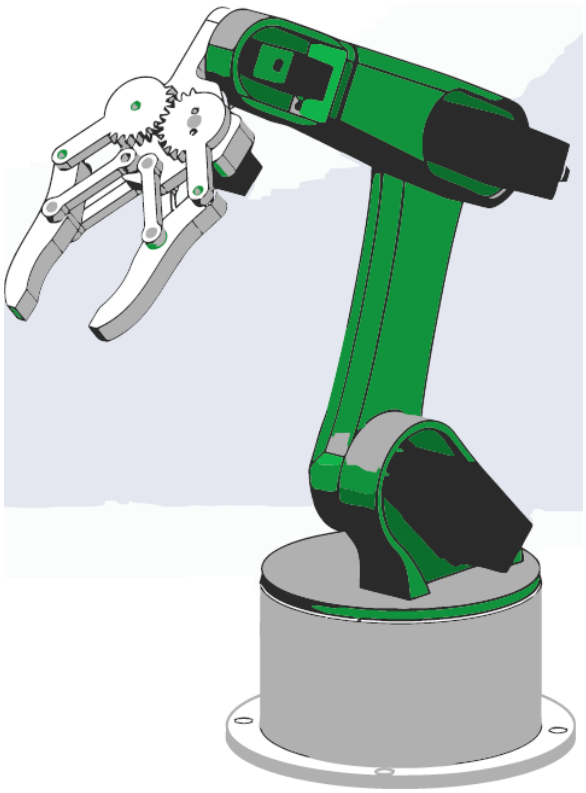
Ahora bien, diseñada las piezas y listas para ser materializadas, se deben extraer de SOLIDWORKS en formato STL, para que el software CURA sea capaz de leer estas piezas y poder configurar la impresora 3d que en este caso es la ENDER 3 PRO, la figura 2.10 muestra cómo se ubican en CURA, para allí imprimirlas a:

- 40% de relleno
- Generar soporte en ángulos mayores a 50
- Imprima las piezas en calidad DYNAMIC QUALITY 0.16mm
- Genere un espesor de base de 2mm
- Extrusor a 200 grados
- Cama caliente a 60 grados

Para un total de 35 horas de impresión, 280 gramos de material se imprimen todas las piezas en PLA.
Ilustración 23 Ensamblaje final del brazo en SOLIDWORKS

Construcción

Para esta se tienen tres ítems, la primera, la construcción del prototipo, la segunda



es la construcción del circuito y la ultima y no menos importante, la programación.

Construcción Mecánica

Todo comienza con el alistamiento de las piezas impresas, estas salen con soportes, los cuales toca retirar, dejar que las piezas se separen de la cama caliente.

Después de estos pasos, se procedió al ensamblaje de los servos en las correspondientes piezas. Para las tres primeras piezas, que son las piezas donde se reposa todo el peso del brazo, se utilizaron los servomotores de piñoneria metálica (base, hombro, codo), esto debido a su torque y par mayor a los micros. Para las tres articulaciones restantes se utilizaron los micro servos, debido a que no debe existir mucho peso en el extremo del brazo o si no la estructura sufre altos valores de torque; Además, las cargas que presenta el brazo no son tan altas, por lo que hace a estos servos ideales para el trabajo.

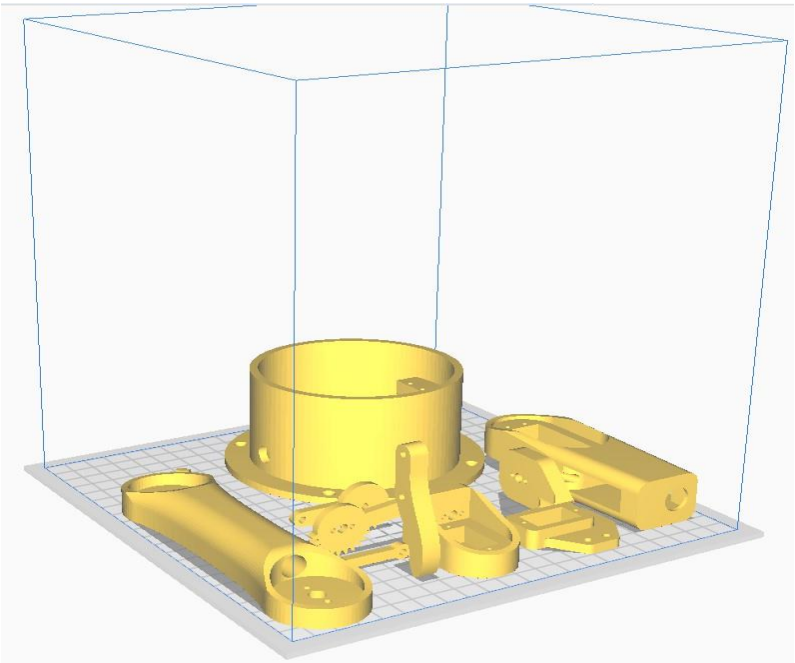


Ilustración 24 Piezas montadas en CURA

Seguido, se fija la base del brazo a una tabla que sirve de base para el proyecto en general. Para después ver el ensamblaje final como en la imagen siguiente.

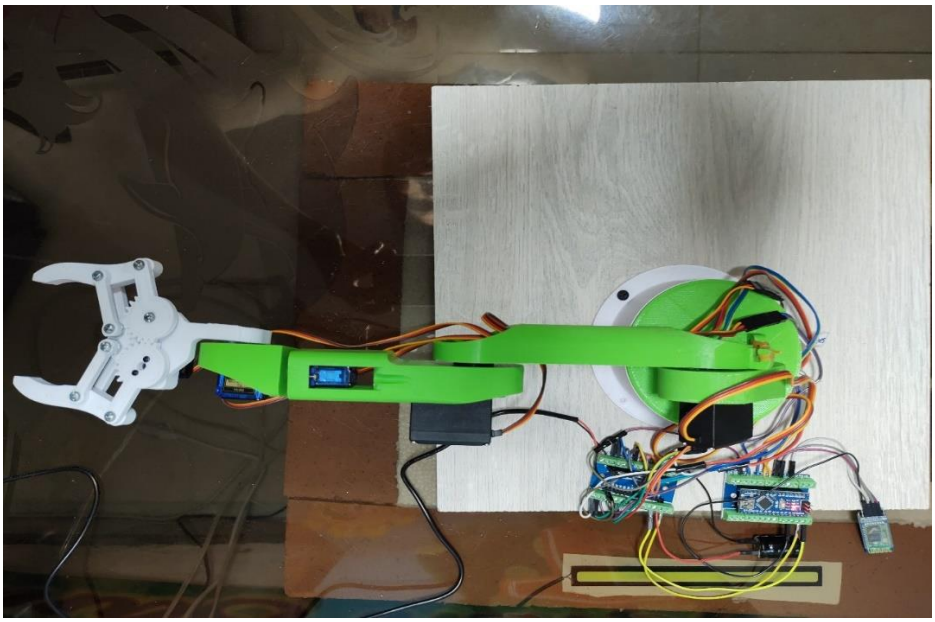


Ilustración 25 Ensamble Final

Construcción Circuito

Como se puede observar en la figura 1.5, el circuito es montado en la protoboard, teniendo dos fuentes de alimentación, la primera es la alimentación del Arduino uno (6 volts-1 amp), la segunda, una fuente de alimentación de 5 voltios a 3 amperios, ésta última encargada de proporcionar la energía a los seis servos.

Teniendo el circuito montado, se procede probar continuidad con el multímetro, antes de encender las fuentes para no llegar a generar ningún daño.

Como opción de mejoramiento, se puede realizar el circuito en PCB, esto quiere decir realizar el circuito impreso en una váquela y soldar los diferentes componentes. Esto con el fin de hacer el proyecto más presentable y disminuir ruidos que la protoboard puede generar.

Programación

En esta sección hay dos partes, la programación para el Arduino, todas las funciones que tenga el brazo sean ejecutadas por este código, y como segunda parte, la programación de la aplicación móvil creada por MIT APPINVENTOR,

encargada de ser la interfaz Humano Maquina, esta última encargada de enviar los datos al módulo bluetooth.

Programación Arduino

Con extensión ".*ino"* se tiene un archivo más de 298 líneas de código en el cual se ejecutan diferentes comandos, librerías, ciclos y demás herramientas que hacen el funcionamiento del brazo real.

Iniciando el programa se ve como se incluyen dos librerías, uno de los servos y la otra de la comunicación serial. Con estas se establecen los nombres de los diferentes servos, su posición inicial, y se establecen los pines 3 y 4 como tx y rx. En la sección de configuración, es necesario inicializar los servos y el módulo Bluetooth y mover el brazo del robot a su posición inicial. Se ejecuto usando la función *write* () que simplemente mueve el servo a cualquier posición de 0 a 180 grados. En la sección de *loop*, utilizando la función Bluetooth. *available* (), se comprueba constantemente si hay algún dato entrante desde el Smartphone. Si es verdadero, usando la función *readString* () esta lee los datos como una cadena y los almacena en la variable *dataIn.* Dependiendo de los datos recibidos, le ordena al brazo robótico qué hacer. Como anexo, se entrega el archivo formato ino.

Programación Aplicativo

Para este paso se utilizó la aplicación en línea MIT App Inventor. En la parte superior se cuenta con dos botones para conectar el smartphone al módulo Bluetooth HC-05. Luego, en el lado izquierdo hay una imagen del brazo del robot, y en el lado derecho se tiene el deslizador de seis para controlar los servos y un deslizador para el control de velocidad.

Cada control deslizante tiene un valor inicial, mínimo y máximo diferente que se adapta a las articulaciones del brazo del robot. En la parte inferior de la aplicación están presentes tres botones, GUARDAR, EJECUTAR y RESTABLECER a través de los cuales permiten programar el brazo robótico para que se ejecute automáticamente. También hay una etiqueta debajo que muestra la cantidad de pasos que han sido guardados.

Ilustración 26 Construcción del Aplicativo

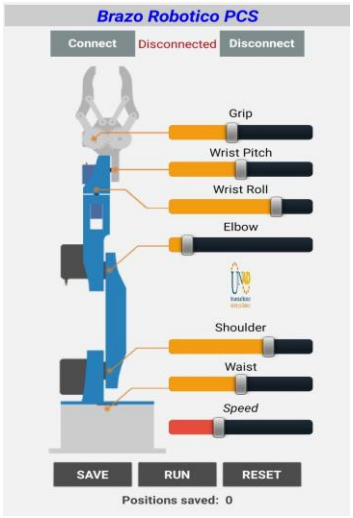


Ilustración 27 Aplicativo Móvil

RESULTADOS Y MÉTODO

Eslabones (También llamado enlace, vinculo o uniones)**:**

Cada elemento rígido y que da la estructura física del robot manipulador está compuesto por PLA (Poliácido Láctico) que fue elegido por ser un material biodegradable proveniente de la descomposición del maíz, entre otras materias primas renovables. Las piezas fueron impresas por medio de la empresa Fused Form con filamento PLA de 1.75 mm, con piezas verdes y otras blancas de 0,4 de relleno y con capas de 0,15 mm, lo anterior suficiente para darle la fortaleza y el peso ideal para el trabajo que realiza.

En la siguiente tabla se relacionan las piezas estructurales impresas ben 3D del brazo robótico:

Tabla 11 las piezas estructurales impresas en 3D

#	Cantidad	Eslabón		Color	Medidas
1	1	Base		Blanco	Alto: 5.7 cm Ancho: 9.7 cm
2	1	Hombro		Verde	Alto: 6.5 cm Ancho: 9.7 cm
3	1	Brazo		Verde	Largo: 15.88 Ancho: 2.2 cm
4	1	Antebrazo		Verde	Largo; 11.7 cm Ancho: 2.7 cm
5	1	Brida o muñeca		Blanco	Largo: 4.5 Ancho: 2.8 cm
6	1	Pinza	Base pinzas	Blanco	Largo: abierta 11.5 cm y cerrada 14.2 Ancho: abierta 8.3 y cerrada 2.6 cm
	1		Engranaje efector pinza activo	Blanco	
	1		Engranaje paralelo pasivo	Blanco	
	4		Estabilizadores de pinzas	Blanco	
	2		Dedos de las pinzas	Blanco	
Total	14 Uds.				

Autoría: propia

Tabla 12 Especificaciones:

Especificaciones del brazo robot	
Estructura mecánica	Articulación vertical
Número de ejes	6 con la pinza
Rango del eje	▪ Eje 1: Rotación de la base o cintura: 180° ▪ Eje 2: Rotación del hombro: 180°

	▪ Eje 3: Rotación del codo: 180° ▪ Eje 4: Rotación de la muñeca: 150° ▪ Eje 5: Inclinación de la muñeca: ▪ Eje 6: Cierre de la pinza: 90°
Radio máximo de trabajo	430 mm (16.929”) Aprox.
Efector final	Servopinza eléctrica de DC. Pinza doble soporte móvil de impulsión directa. http://www.sapiensman.com/tecnoficio/soldadura/soldadura robotica1.php
Retroalimentación	Codificadores óhmicos de 180° (Potenciómetros)
Actuadores	Servomotores MG996R en los tres primeros ejes. Micro servomotores SG90 para los 3 ejes restantes contando la pinza. Todos a 5VDC
Control de los actuadores	Analógica por PWM 20 ms ó 50 Hz (Modulación de ancho de pulso)
Transmisión	Directa por medio de acoples plásticos
Carga máxima	5 kg (11 lbs.) (Con velocidad mínima)
Peso	1 kg Aproximadamente
Repetibilidad de la posición	+-0,6 mm
Temperatura de operación	0° a 39°C

Autoría: propia

Estructura

Este prototipo es un robot de articulación vertical, posee 5 articulaciones de revolución que sumando la pinza posee cinco grados de libertad. El diseño del brazo robótico facilita al efector final la orientación y la posición en cualquier punto dentro de un gran espacio de operación con que cuenta.
Todas las articulaciones de este prototipo son maniobradas por servomotores de C.C que integran una caja mecánica reductora y cuyos elementos están conectados directamente a cada eslabón por medio de acoples plásticos en sus ejes.

Tabla 13 articulaciones

Nº de eje	Nombre de la articulación	Movimiento	Nº de motor
1	Base o cintura	Gira el cuerpo	1

2	Hombro	Sube y baja el brazo	2
3	Codo	Sube y baja el antebrazo	3
4	Rotación Muñeca	Gira el efector final	4
5	Inclinación Muñeca	Sube y baja el efector final	5

Autoría: propia

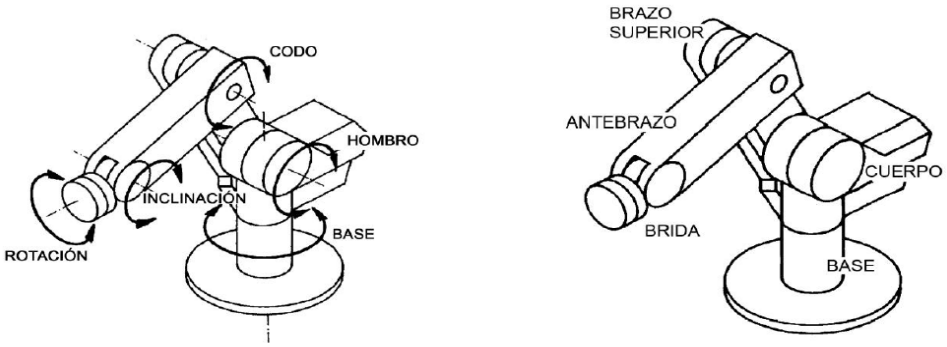


Ilustración 28 Manual Scorbot ER9 Pro,

Tomado de: https://downloads.intelitek.com/Manuals/Robotics/Spanish/Scorbot-ER-9Pro-ES-B.pdf

En la imagen primera se observa la estructura del robot denotando las articulaciones, motores, o simplemente ejes donde se ejerce el movimiento.
En la segunda imagen hace referencia a la misma estructura del robot, pero enmarcando los eslabones.
Para las anteriores imágenes se basó en el Scorbot ER Pro para hacer referencia a la tabla anterior de manera gráfica.

Descripción mecánica de la pinza:

La pinza es accionada de manera directa por un micro servomotor eléctrico C.C con ayuda de otras piezas que componen esta herramienta manipuladora y le brindan una mayor estabilidad.
El mecanismo de la pinza es paralelogramo articulado de dos puntos fijos (P_1 y P_2) y dos puntos móviles que para una mejor compresión se exponen dos imágenes:

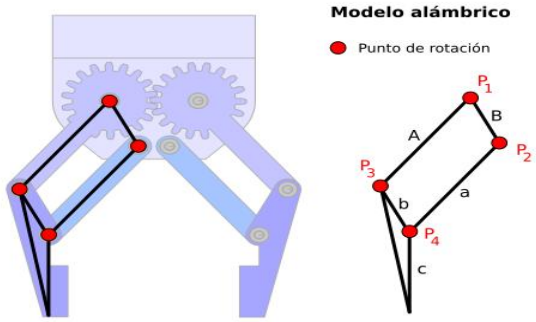


Ilustración 29 mecanismos de la pinza 1

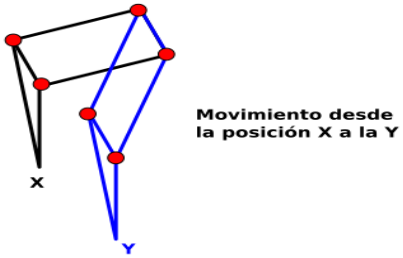


Ilustración 30 mecanismos de la pinza 2

Las imágenes fueron tomadas del siguiente sitio web, y el contenido anterior sobre el mecanismo de la pinza fueron basado de: http://diwo.bq.com/builds/pinza-robotica-porta-acreditaciones/

Capacidad de agarre

La pinza puede tener una apertura máxima de 5.7 cm y un cierre total.

Tabla 14 motores

Especificaciones para la articulación 1, 2 y 3	
Referencia	MG996R Tower Pro
Denominación por tamaño	Servo
Peso	55 g
Stall torque	9.4 kgf·cm (4.8 V)
Caja reductora	Metálica
Dimensión	40.7 x 19.7 x 42.9 mm aprox.

Modulación de control	Analógica PWM
Velocidad de funcionamiento	0.17 s/60º (4.8 V)
Ciclo por pulso	ca. 20 ms (50 Hz)
Voltaje de funcionamiento	4,8 V (~ 5 V)
Rango de temperatura:	0 ºC - 55 ºC
Rango rotacional	180°
Rango de temperatura	0° a 55°C
Conexión eléctrica	Rojo: Positivo (+) Café: Negativo (-) Naranja: Señal
Conector	Hembra tipo 'S' de 3 pines

Autoría propia

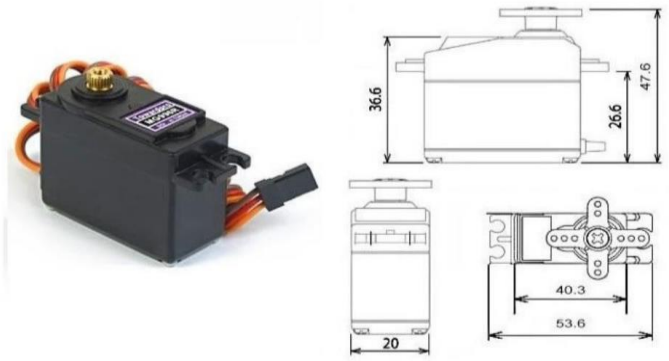


Ilustración 31 servo

Tomado de:
https://pdf1.alldatasheet.es/datasheet-pdf/view/1131873/ETC2/MG996R.html

Tabla 15 motores

Especificaciones para la articulación 4, 5 y 6	
Referencia	SG90 Tower Pro
Denominación por tamaño	Microservo
Peso	9 g
Stall torque	1.5 kg/cm a 4.8V
Caja reductora	Plástica
Dimensión	22,2 x 11,8 x 31 mm aprox.
Modulación de control	Analógica PWM
Velocidad de funcionamiento	0,1 s / 60° a 4.8V

Ciclo por pulso	ca. 20 ms (50 Hz)
Ancho de pulso	500 a 2400 µs
Voltaje de funcionamiento	4,8 V (~ 5 V)
Ancho de banda muerta	10 µs
Rango de temperatura:	0 ºC - 55 ºC
Rango rotacional	180°
Rango de temperatura	0° a 55°C
Conexión eléctrica	Rojo: Positivo (+) Café: Negativo (-) Naranja: Señal
Conector	Hembra tipo 'S' de 3 pines

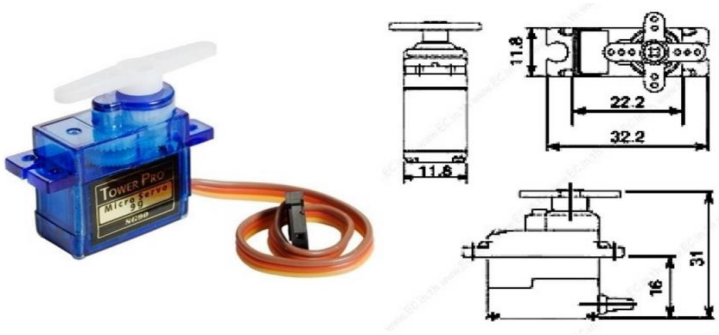


Ilustración 32 servo

Tomado de: https://datasheetspdf.com/datasheet/SG90.html

Módulo Bluetooth:

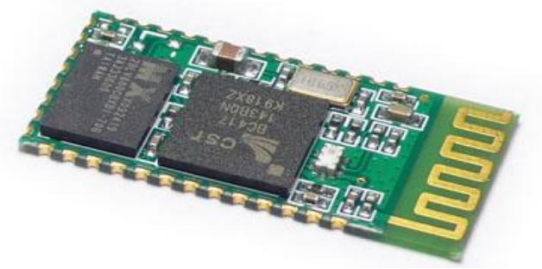


Ilustración 33 Bluetooth

Tabla 16 Bluetooth características

Características	
Tamaño	12,7 mm x 27 mm
Versión	2.0 + EDR (2004)
Ancho de banda (BW)	3 Mbps

Conexión	6 pines
Banda espectro electromagnético	2.4 GHz
Modelo de comunicación (ROLL)	Esclavo/maestro
Voltaje	5V
Sensibilidad típica	-80dBm
Potencia de transmisión	4 dBm de potencia de transmisión RF
Antena	Integrada
Interfaz UART	Programable de 1200 a 1382400

Comandos AT del HC-05			
Comando	**Parámetro**	**Configuración por defecto**	**Configuración necesaria**
AT	Verifica la conexión y responde OK		
AT+NAME	Muestra el nombre del modulo	HC-05	
AT+NAME=(xxxx)	Cambia Nombre		Prototipo Scorbot
AT+UART	Muestra la velocidad de transmisión de datos en baudios	9600	
AT+UART=xxxx,0,0	Modifica los baudios		38400
AT+ROLE	Muestra el rol del módulo (AT+ROLE=0)	Esclavo	
AT+ROLE= 1 ó 0	Establece el rol seleccionado		Esclavo
AT+PSWD	Permite ver la contraseña	1234	
AT+PSWD=xxxx	Modifica la contraseña		1234

Tabla 17 Comandos AT del HC-05

Tomado de https://datasheetspdf.com/pdf-file/1418730/ITead/HC-05/1

Arduino nano Atmega 328P

Se determino que en este proyecto lo más económico, suficiente pequeño, congruente con el tamaño del prototipo, asequible, exequible es el Arduino Nano, ya que cuenta con las características de conexión perfectas para trabajar de acuerdo con las necesidades del brazo robótico.

Ilustración 34 Imágenes ilustrativas de la placa usada.

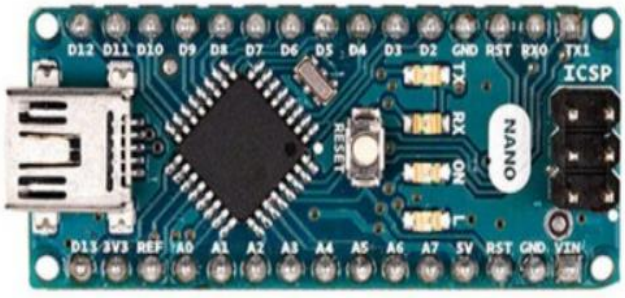


Ilustración 35 Imágenes ilustrativas de la placa usada.

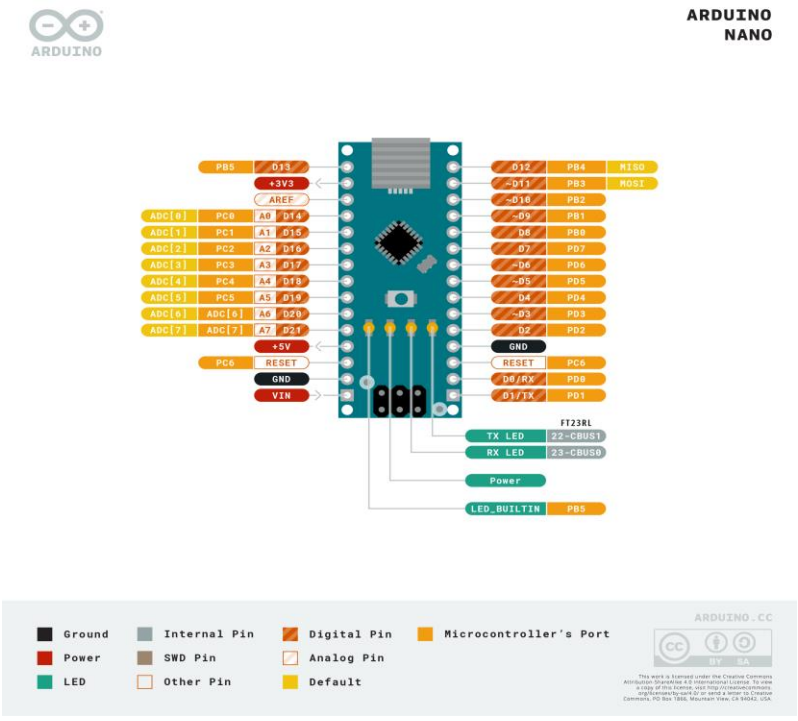


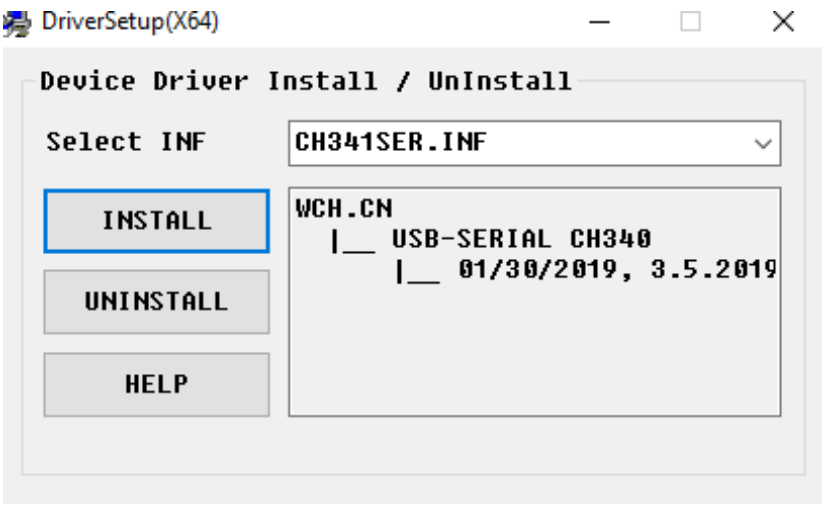
Ilustración 36 Esquemático eléctrico de los puertos y pines del Arduino Nano

Imágenes tomadas de: https://store.arduino.cc/usa/arduino-nano

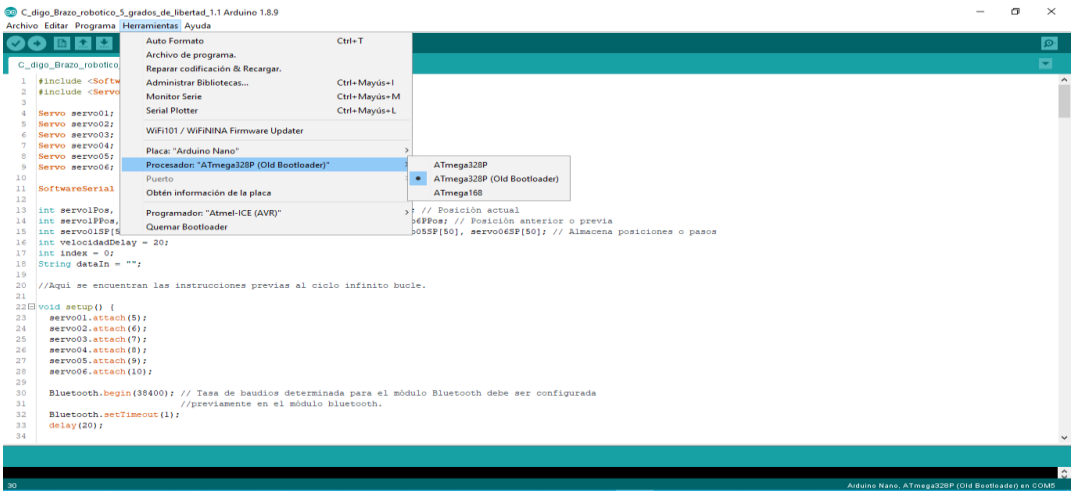
Esta placa de desarrollo posee las salidas PWM que se necesitan para controlar los servomotores, también permite configurar los pines TX y RX para la comunicación con el Bluetooth HC-05.

En total fueron usados 8 puertos vitales, 6 con salida de la señal para el control con PWM y dos configuradas para configuración serial TX-RX.

Es importante aclarar que el Arduino usado en este proyecto es genérico, por tanto, fue imprescindible descargar el controlador llamado del dispositivo llamado “CH341SER” e instalarlo para el reconocimiento como dispositivo en el computador de trabajo. Además se tuvo que seleccionar en el programa de Arduino el procesador ATmega328P (*Old Booloader*), ya que por la versión de software Arduino 1.8.9, era muy nueva en comparación a la versión de placa Arduino, de esta manera se logró cargar el código al Nano.



Esta imagen anterior muestra el momento de la instalación del controlador del Arduino Nano genérico.



Esta imagen anterior muestra el momento en que se seleccionó la versión de procesador para un buen reconocimiento del software Arduino 1.8.9.

Diagrama eléctrico

Este esquema eléctrico en sí es sencillo pero elemental para entender la configuración y conexión de los dispositivos que intervienen en este proyecto.
Para diseñarlo fue necesario instalar el software PROTEUS 8 Profesional y las librerías de bluetooth HC-05 y el Arduino Nano Atmega 328P.

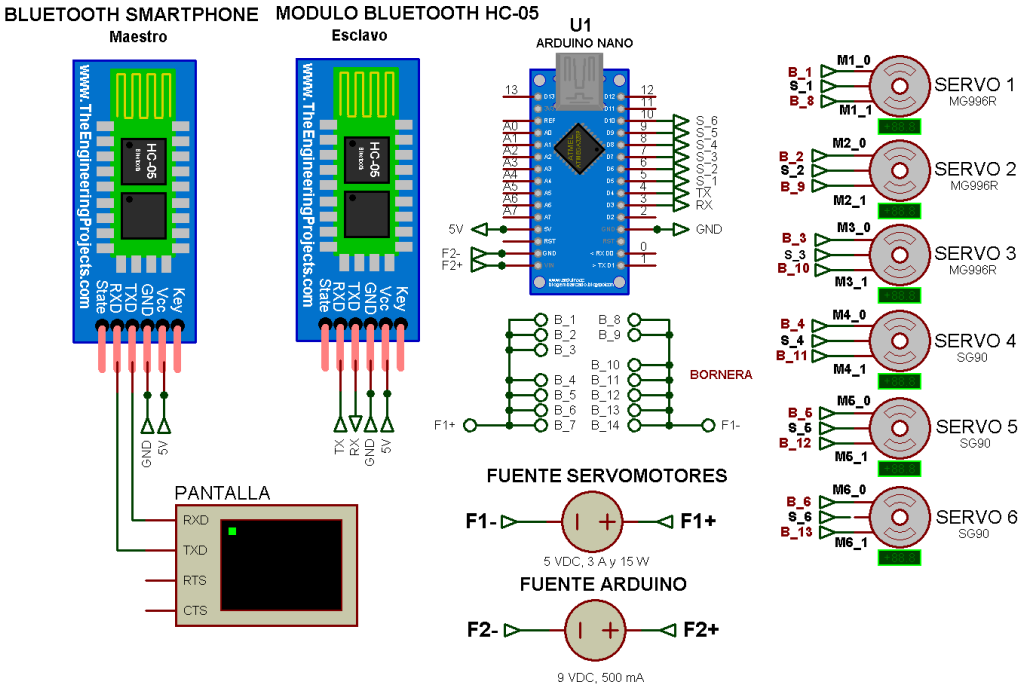


Ilustración 37 esquemático

Enlace del proyecto en el software Proteus del esquemático:
https://drive.google.com/file/d/1IGdiHGYv3U5jpefBr-0zeuANtzIWwe8_/view?usp=sharing

Se utilizo una bornera para la conexión de todos los positivos y negativos de cada una de las alimentaciones requeridas por los servomotores, bluetooth y el mismo Arduino Nano.
También se implementó una base o adaptador con borneras para la conexión sencilla y practica de cada una de las conexiones, con el fin de que sea práctico a los aprendices que quieran ahondar los temas de robótica incluyendo el ensamble de este en la parte eléctrica.

Cuadro detallado del cableado y conexiones (Alimentación y señal de control:

Tabla 18 cableado y conexiones

Eje	N.° de pin del cableado	Descripción del pin del robot	Color de cable Belden	Descripción del pin del controlador y bornera
1	M1_0	Servomotor 1 +	Rojo	B_1
	M1_1	Servomotor 1-	Café	B_8
	S_1	Señal Servomotor 1	Naranja	D5
2	M2_0	Servomotor 2 +	Rojo	B_2
	M2_1	Señal Servomotor 2	Café	B_9
	S_2	Servomotor 2	Naranja	D6
3	M3_0	Servomotor 3 +	Rojo	B_3
	M3_1	Servomotor 3 -	Café	B_10
	S_3	Señal Servomotor 3	Naranja	D7
4	M4_0	Micro Servo 4 +	Rojo	B_4
	M4_1	Micro Servo 4 -	Café	B_11
	S_4	Señal Micro Servo 4	Naranja	D8
5	M5_0	Micro Servo 5 +	Rojo	B_5
	M5_1	Micro Servo 5 -	Café	B_12
	S_5	Señal Micro Servo 5	Naranja	D9
6	M6_0	Micro Servo 6 +	Rojo	B_6
	M6_1	Micro Servo 6 -	Café	B_13
	S_6	Señal Micro Servo 6	Naranja	D10
Elemento	**N.° de pin del cableado**	**Descripción del pin del robot**	**Color de cable Belden**	**Descripción del pin del controlador y bornera**

Modulo inalámbrico	BT_0	Bluetooth +	Negro	5V
	BT_1	Bluetooth -	Gris oscuro	GND
	BT	TX	Gris claro	D4
	BX	RX	Morado	D3
Fuente Servomotores	F1+	Fuente uno +	Rojo	B_7
	F1-	Fuente uno -	Negro	B_14
Fuente Arduino	F2+	Fuente dos +	Rojo	Vin
	F2-	Fuente dos-	Negro	GND

Autoría: propia

Código Arduino

Lo verdaderamente complejo se demuestra en el código de Arduino que es el que hace posible la recepción de los valores de la APP y otorga el movimiento de cada articulación conforme a lo solicitado.
Son tres variables importantes y elementales para llevar a cabo el modo manual de control del brazo robótico, una es la de velocidad de los movimientos de los servo, la otra la que controla la cantidad de giro o la posición, las demás variables son complementarias que permiten que las elementales puedan cumplir su función y la última variable del proyecto es la que admite almacenar y grabar los movimientos, es decir, cada posición de cada servo para determinar un proceso infinito el cual es denominada modo automático.
Se tuvo en cuenta los movimientos límites para cada articulación para evitar choques fiscos entre los es, para ello se configuro desde la programación de circuito.

A continuación, se dará explicación al código paso a paso:

Aquí se declaran las variables e incluyen las librerías necesarias para la comunicación serial con el bluetooth junto con la de los Servomotores.
Se debe tener en cuenta que previamente se configuro el módulo Bluetooth HC-05 donde se especificó la velocidad en baudios en la comunicación y el nombre.

```
#include <SoftwareSerial.h>
#include <Servo.h>
```

Se declara cada variable de los servomotores 1 al 3 son los MMG996R y del motor 4 al 6 son los SG90, también el módulo bluetooth y de almacenamiento de valores de posición.

```
Servo ServoM01;
Servo ServoM02;
Servo ServoM03;
```

```
    Servo MicroServoM04;
    Servo MicroServoM05;
    Servo MicroServoM06;

SoftwareSerial Bluetooth(3, 4); // Arduino (RX, TX) - HC-05 Bluetooth (TX, RX)

int ServoM1_Pos, ServoM2_Pos, ServoM3_Pos, MicroServoM4_Pos, MicroServoM5_Pos, MicroServoM6_Pos; // Posición actual
int Servo1_Pos_Prev, Servo2_Pos_Prev, Servo3_Pos_Prev, MicroServo4_Pos_Prev, MicroServo5_Pos_Prev, MicroServo6_Pos_Prev; // Posición anterior o previa
int ServoM01_POS_G[50], ServoM02_POS_G[50], ServoM03_POS_G[50], MicroServoM04_POS_G[50], MicroServoM05_POS_G[50], MicroServoM06_POS_G[50]; // Almacena posiciones / pasos
int velocidadDelay = 30;
int index = 0;
String dataIn = "";
```

Aquí se encuentran las instrucciones previas al ciclo infinito bucle, en esta sección del código se dan las instrucciones iniciales al bazo robot, y se declaran variables para almacenar posiciones de los motores.

```
void setup() {
    ServoM01.attach(5); // Este Motor se encarga de mover la Cintura del brazo robótico.
    ServoM02.attach(6); // Este Motor se encarga de mover el Hombro del brazo robótico.
    ServoM03.attach(7); // Este Motor se encarga de mover el Codo del brazo robótico.
    MicroServoM04.attach(8); // Este Motor se encarga de girar la muñeca del brazo robótico, también
        // se conoce como brida
    MicroServoM05.attach(9); // Este Motor se encarga de inclinar/declinar, la muñeca del brazo robótico.
    MicroServoM06.attach(10);// Este Motor se encarga de cerrar y abrir la pinza, también se le llama
        // agarre.

    Bluetooth.begin(38400); // Tasa de baudios determinada para el módulo Bluetooth debe ser configurada previamente en el módulo bluetooth.
    Bluetooth.setTimeout(1);
    delay(30);
```

Posición previa (también inicial o de reposo) del brazo robótico: Antes de ir a void loop se dan las siguientes instrucciones, también llamado "Home".

Se debe tener en cuenta que se definieron unos límites en grados de algunos servomotores para evitar choques físicos que comprometan el hardware.
Cabe especificar los rangos posibles por los servomotores es de 0° a 180°

```
Servo1_Pos_Prev = 90; // En contra de las manecillas del reloj cuanto más valor
        // El brazo va hacia la derecha entre más valor e izquierda en caso
        //contrario
 ServoM01.write(Servo1_Pos_Prev); // Cintura (Rango límite 0° a 180°)
 delay(700);

 Servo2_Pos_Prev = 150; // Cuanto más valor, sube el brazo y va hasta atrás
 por medio de la articulación del hombro, en contra de las manecillas del reloj
 ServoM02.write(Servo2_Pos_Prev); // Hombro (Rango límite 0° a 180°)
 delay(700);

 Servo3_Pos_Prev = 120; // Entre más valor va hacia abajo el antebrazo y
 entre menos valor hacia arriba
 ServoM03.write(Servo3_Pos_Prev); // Codo (Rango límite 0° a 150°)
 delay(700);

 MicroServo4_Pos_Prev = 90; // Entre más valor la muñeca gira a la izquierda
 y viceversa
 MicroServoM04.write(MicroServo4_Pos_Prev); // Rotación de Muñeca
 (Rango límite 0° a 180°)
 delay(700);

 MicroServo5_Pos_Prev = 60; //Cuanto más valor la inclinación es más hacia
 arriba
 MicroServoM05.write(MicroServo5_Pos_Prev); // Inclinación de Muñeca
 (Rango límite 0° a 180°)
 delay(700);

 MicroServo6_Pos_Prev = 140; // Cuanto más valor abre proporcionalmente
 la pinza
 MicroServoM06.write(MicroServo6_Pos_Prev); // Pinza o agarre (Rango
 límite 60° a 150°)
 delay(300);
```

Se hace un saludo abriendo y cerrando las pinzas:

```
MicroServo6_Pos_Prev = 60;
MicroServoM06.write(MicroServo6_Pos_Prev);// Pinza o agarre
delay(300);
MicroServo6_Pos_Prev = 140;
MicroServoM06.write(MicroServo6_Pos_Prev);// Pinza o agarre
```

```
delay(300);
MicroServo6_Pos_Prev = 60;
MicroServoM06.write(MicroServo6_Pos_Prev);// Pinza o agarre
delay(300);
MicroServo6_Pos_Prev = 140;
MicroServoM06.write(MicroServo6_Pos_Prev);// Pinza o agarre
delay(300);
MicroServo6_Pos_Prev = 60;
MicroServoM06.write(MicroServo6_Pos_Prev);// Pinza o agarre
delay(300);
MicroServo6_Pos_Prev = 140;
MicroServoM06.write(MicroServo6_Pos_Prev);// Pinza o agarre
delay(300);
}
```

Aquí empieza el ciclo infinito de repeticiones:

```
void loop() {

// Verifica datos entrantes:

 if (Bluetooth.available() > 0) {
 dataIn = Bluetooth.readString(); // Lee los datos se almacenan como variable
cadena

Si el control deslizante "Cintura" ha cambiado de valor, mueva el Servomotor
1 a la posición:

 if (dataIn.startsWith("s1")) {
  String dataInS = dataIn.substring(2, dataIn.length()); // Extrae solo el
número. E.g. de "s1120" a "120"

  ServoM1_Pos = dataInS.toInt(); // Convierte la cadena en variable entero

// Se usan bucles for para poder controlar la velocidad del servo
// Si la posición anterior es mayor que la posición actual

  if (Servo1_Pos_Prev > ServoM1_Pos) {
  for ( int j = Servo1_Pos_Prev; j >= ServoM1_Pos; j--) { // Acciona el
Servomotor hacia abajo
   ServoM01.write(j);
   delay(30); // Define la velocidad a la que gira el Servomotor
  }
  }
```

Si la posición anterior es más pequeña que la posición actual:

```
 if (Servo1_Pos_Prev < ServoM1_Pos) {
 for ( int j = Servo1_Pos_Prev; j <= ServoM1_Pos; j++) { // Acciona el Servomotor hacia arriba
  ServoM01.write(j);
  delay(30);
 }
 }
 Servo1_Pos_Prev = ServoM1_Pos; // Establecer la posición actual como la posición previa o anterior
}
```

Mueve el Servomotor 2:

```
if (dataIn.startsWith("s2")) {
 String dataInS = dataIn.substring(2, dataIn.length());
 ServoM2_Pos = dataInS.toInt();

 if (Servo2_Pos_Prev > ServoM2_Pos) {
 for ( int j = Servo2_Pos_Prev; j >= ServoM2_Pos; j--) {
  ServoM02.write(j);
  delay(50);
 }
 }
 if (Servo2_Pos_Prev < ServoM2_Pos) {
 for ( int j = Servo2_Pos_Prev; j <= ServoM2_Pos; j++) {
  ServoM02.write(j);
  delay(50);
 }
 }
 Servo2_Pos_Prev = ServoM2_Pos;
}
```

Mueve Servomotor 3:

```
if (dataIn.startsWith("s3")) {
 String dataInS = dataIn.substring(2, dataIn.length());
 ServoM3_Pos = dataInS.toInt();
 if (Servo3_Pos_Prev > ServoM3_Pos) {
 for ( int j = Servo3_Pos_Prev; j >= ServoM3_Pos; j--) {
  ServoM03.write(j);
  delay(30);
 }
 }
```

```
 if (Servo3_Pos_Prev < ServoM3_Pos) {
 for ( int j = Servo3_Pos_Prev; j <= ServoM3_Pos; j++) {
  ServoM03.write(j);
  delay(30);
 }
 }
 Servo3_Pos_Prev = ServoM3_Pos;
}
// Mueve micro Servomotor 4
if (dataIn.startsWith("s4")) {
 String dataInS = dataIn.substring(2, dataIn.length());
 MicroServoM4_Pos = dataInS.toInt();
 if (MicroServo4_Pos_Prev > MicroServoM4_Pos) {
 for ( int j = MicroServo4_Pos_Prev; j >= MicroServoM4_Pos; j--) {
  MicroServoM04.write(j);
  delay(30);
 }
 }
 if (MicroServo4_Pos_Prev < MicroServoM4_Pos) {
 for ( int j = MicroServo4_Pos_Prev; j <= MicroServoM4_Pos; j++) {
  MicroServoM04.write(j);
  delay(30);
 }
 }
 MicroServo4_Pos_Prev = MicroServoM4_Pos;
}
// Mueve micro Servomotor 5
if (dataIn.startsWith("s5")) {
 String dataInS = dataIn.substring(2, dataIn.length());
 MicroServoM5_Pos = dataInS.toInt();
 if (MicroServo5_Pos_Prev > MicroServoM5_Pos) {
 for ( int j = MicroServo5_Pos_Prev; j >= MicroServoM5_Pos; j--) {
  MicroServoM05.write(j);
  delay(30);
 }
 }
 if (MicroServo5_Pos_Prev < MicroServoM5_Pos) {
 for ( int j = MicroServo5_Pos_Prev; j <= MicroServoM5_Pos; j++) {
  MicroServoM05.write(j);
  delay(30);
 }
 }
 MicroServo5_Pos_Prev = MicroServoM5_Pos;
}
```

```
 // Mueve el micro Servomotor 6
 if (dataIn.startsWith("s6")) {
  String dataInS = dataIn.substring(2, dataIn.length());
  MicroServoM6_Pos = dataInS.toInt();
  if (MicroServo6_Pos_Prev > MicroServoM6_Pos) {
  for ( int j = MicroServo6_Pos_Prev; j >= MicroServoM6_Pos; j--) {
   MicroServoM06.write(j);
   delay(30);
  }
  }
  if (MicroServo6_Pos_Prev < MicroServoM6_Pos) {
  for ( int j = MicroServo6_Pos_Prev; j <= MicroServoM6_Pos; j++) {
   MicroServoM06.write(j);
   delay(30);
  }
  }
  MicroServo6_Pos_Prev = MicroServoM6_Pos;
 }

 // Si se presiona el botón "GUARDAR" POS_G: Posición guardada

 if (dataIn.startsWith("GUARDAR")) {

  ServoM01_POS_G[index] = Servo1_Pos_Prev; // Guarda la posición en la
variable Array
  ServoM02_POS_G[index] = Servo2_Pos_Prev;
  ServoM03_POS_G[index] = Servo3_Pos_Prev;
  MicroServoM04_POS_G[index] = MicroServo4_Pos_Prev;
  MicroServoM05_POS_G[index] = MicroServo5_Pos_Prev;
  MicroServoM06_POS_G[index] = MicroServo6_Pos_Prev;
  index++;      // Incrementa el índice del Array
 }

// Si se presiona el botón "ARRANCAR"
 if (dataIn.startsWith("ARRANCAR")) {
  ARRANCARservo(); // Modo automático (ejecuta los pasos guardados)
 }
 // Si se presiona el botón "REINICIAR"
 if ( dataIn == "REINICIAR") {
  memset(ServoM01_POS_G,  0,  sizeof(ServoM01_POS_G));  //  Borrar  los
datos de la Array a 0
  memset(ServoM02_POS_G, 0, sizeof(ServoM02_POS_G));
  memset(ServoM03_POS_G, 0, sizeof(ServoM03_POS_G));
  memset(MicroServoM04_POS_G, 0, sizeof(MicroServoM04_POS_G));
  memset(MicroServoM05_POS_G, 0, sizeof(MicroServoM05_POS_G));
```

```
  memset(MicroServoM06_POS_G, 0, sizeof(MicroServoM06_POS_G));
  index = 0; // Index to 0
 }
 }
}
```

Función personalizada del modo automático (ejecuta los pasos guardados)

```
void ARRANCARservo() {
 while (dataIn != "REINICIAR") { // Ejecuta los pasos una y otra vez hasta que se presione el botón "REINICIAR" (Bucle)
 for (int i = 0; i <= index - 2; i++) { // Ejecuta todos los pasos (index)
  if (Bluetooth.available() > 0) {  // Verifica los datos entrantes
  dataIn = Bluetooth.readString();
  if ( dataIn == "PAUSAR") {   // Si se presiona el botón "PAUSAR"
   while (dataIn != "ARRANCAR") {     // Espera hasta que se presione "ARRANCAR" nuevamente

   if (Bluetooth.available() > 0) {
    dataIn = Bluetooth.readString();
    if ( dataIn == "REINICIAR") {
    break;
    }
   }
   }
  }
  // Si se cambia el control deslizante de velocidad
  if (dataIn.startsWith("ss")) {
   String dataInS = dataIn.substring(2, dataIn.length());
   velocidadDelay = dataInS.toInt(); // Cambiar la velocidad del Servomotor (tiempo de retardo)
  }
  }
```

A continuación, se los cálculos necesarios para definir la nueva posición, esto restando posición anterior o guardada y posición solicitada:

```
  Servomotor 1
  if (ServoM01_POS_G[i] == ServoM01_POS_G[i + 1]) {
  }
  if (ServoM01_POS_G[i] > ServoM01_POS_G[i + 1]) {
  for ( int j = ServoM01_POS_G[i]; j >= ServoM01_POS_G[i + 1]; j--) {
   ServoM01.write(j);
   delay(velocidadDelay);
  }
  }
  if (ServoM01_POS_G[i] < ServoM01_POS_G[i + 1]) {
  for ( int j = ServoM01_POS_G[i]; j <= ServoM01_POS_G[i + 1]; j++) {
```

```
 ServoM01.write(j);
 delay(velocidadDelay);
}
}

Servomotor 2
if (ServoM02_POS_G[i] == ServoM02_POS_G[i + 1]) {
}
if (ServoM02_POS_G[i] > ServoM02_POS_G[i + 1]) {
for ( int j = ServoM02_POS_G[i]; j >= ServoM02_POS_G[i + 1]; j--) {
 ServoM02.write(j);
 delay(velocidadDelay);
}
}
if (ServoM02_POS_G[i] < ServoM02_POS_G[i + 1]) {
for ( int j = ServoM02_POS_G[i]; j <= ServoM02_POS_G[i + 1]; j++) {
 ServoM02.write(j);
 delay(velocidadDelay);
}
}

Servomotor 3
if (ServoM03_POS_G[i] == ServoM03_POS_G[i + 1]) {
}
if (ServoM03_POS_G[i] > ServoM03_POS_G[i + 1]) {
for ( int j = ServoM03_POS_G[i]; j >= ServoM03_POS_G[i + 1]; j--) {
 ServoM03.write(j);
 delay(velocidadDelay);
}
}
if (ServoM03_POS_G[i] < ServoM03_POS_G[i + 1]) {
for ( int j = ServoM03_POS_G[i]; j <= ServoM03_POS_G[i + 1]; j++) {
 ServoM03.write(j);
 delay(velocidadDelay);
}
}

// Micro Servomotor 4
if (MicroServoM04_POS_G[i] == MicroServoM04_POS_G[i + 1]) {
}
if (MicroServoM04_POS_G[i] > MicroServoM04_POS_G[i + 1]) {
for ( int j = MicroServoM04_POS_G[i]; j >= MicroServoM04_POS_G[i + 1];
j--) {
 MicroServoM04.write(j);
 delay(velocidadDelay);
```

```
  }
  }
  if (MicroServoM04_POS_G[i] < MicroServoM04_POS_G[i + 1]) {
  for ( int j = MicroServoM04_POS_G[i]; j <= MicroServoM04_POS_G[i + 1];
j++) {
   MicroServoM04.write(j);
   delay(velocidadDelay);
  }
  }

  // Micro Servomotor 5
  if (MicroServoM05_POS_G[i] == MicroServoM05_POS_G[i + 1]) {
  }
  if (MicroServoM05_POS_G[i] > MicroServoM05_POS_G[i + 1]) {
  for ( int j = MicroServoM05_POS_G[i]; j >= MicroServoM05_POS_G[i + 1];
j--) {
   MicroServoM05.write(j);
   delay(velocidadDelay);
  }
  }
  if (MicroServoM05_POS_G[i] < MicroServoM05_POS_G[i + 1]) {
  for ( int j = MicroServoM05_POS_G[i]; j <= MicroServoM05_POS_G[i + 1];
j++) {
   MicroServoM05.write(j);
   delay(velocidadDelay);
  }
  }

  // Micro Servomotor 6
  if (MicroServoM06_POS_G[i] == MicroServoM06_POS_G[i + 1]) {
  }
  if (MicroServoM06_POS_G[i] > MicroServoM06_POS_G[i + 1]) {
  for ( int j = MicroServoM06_POS_G[i]; j >= MicroServoM06_POS_G[i + 1];
j--) {
   MicroServoM06.write(j);
   delay(velocidadDelay);
  }
  }
  if (MicroServoM06_POS_G[i] < MicroServoM06_POS_G[i + 1]) {
  for ( int j = MicroServoM06_POS_G[i]; j <= MicroServoM06_POS_G[i + 1];
j++) {
   MicroServoM06.write(j);
   delay(velocidadDelay);
  }
  }
```

```
}
}
}
```

APP de control a distancia

La aplicación “Control de Brazo Robótico UNAD” orientado para sistemas operativos Android fue diseñada y codificada en MIT APP Inventor de Google Labs.
App Inventor es una plataforma Web la cual permite en línea crear proyectos aplicativos para celulares Android a través de programación de alto nivel, concretamente por medio de conexión entre bloques ya definidos por la plataforma, con el fin de que el usuario pueda unir y formar una serie de algoritmos a conveniencia para lograr algún proceso especifico requerido, en este caso, el control de 6 servomotores del manipulador robótico. Además de la parte lógica, también posee una interfaz de diseño de la App, con la cual interactúa smartphone y el estudiante en este caso, siendo de la siguiente manera:

Paleta de diseño de la APP en App Inventor:

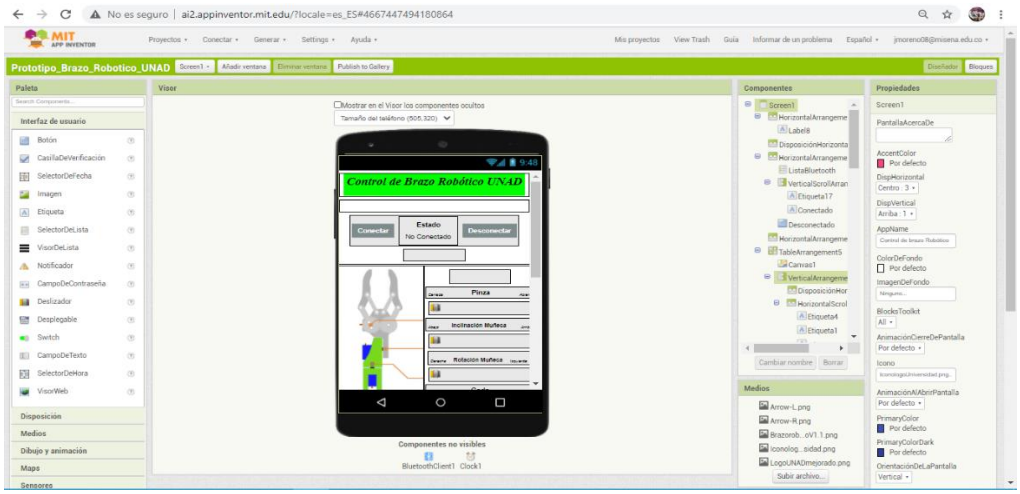


Ilustración 38 diseño de la APP en App Inventor

Fuente de la plataforma para el desarrollo: http://ai2.appinventor.mit.edu/

Apariencia desde dentro de la APP:

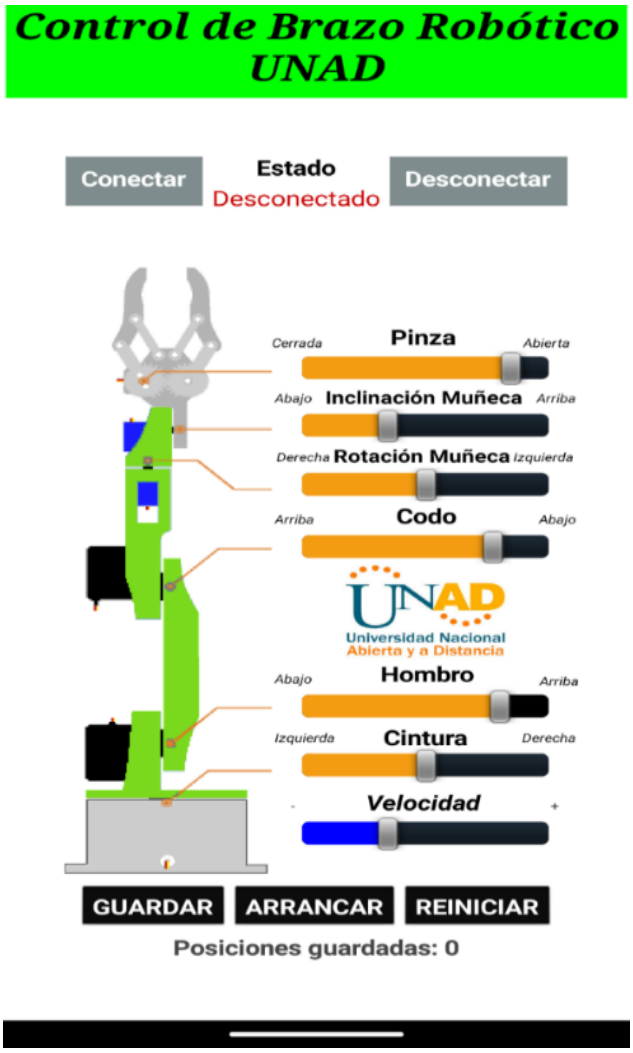


Ilustración 39 App Desde un smartphone Motorola G9 PLUS

El anterior pantallazo fue tomado desde un teléfono inteligente Motorola G9 PLUS al cual se le instalo la aplicación.
En esta aplicación el usuario manipula unas variables que serán enviadas por bluetooth por medio de un smartphone hacia Arduino con el fin de controlar la magnitud de movimiento con velocidad, la cantidad de movimiento por cada servo y la opción de guardar los pasos para ejecutar automáticamente un proceso con el brazo robótico.

Enlace de la APP "Prototipo_Brazo_Robotico_UNAD.apk":

Código QR

Enlace APK:
https://drive.google.com/file/d/13t76TT8t25pdEpVfEvEdQXZjg7Pv7tOf/view?usp=drivesdk

Funcionamiento general de la APP:

Ilustración 40 Funcionamiento general de la APP

En la parte de arriba hay dos botones, uno izquierdo para conectar a un dispositivo bluetooth, en este caso es con el módulo HC-05, y otro al lado derecho uno de desconexión. Debe tenerse en cuenta que debe estar sincronizado desde la herramienta Bluetooth del teléfono y obviamente activado el Bluetooth. Se da clic en conectar y elige el nombre del bluetooth HC-05 que está configurado como PROTOTIPO SCORBOT.

En el medio de los botones mencionados anteriormente se encuentra un indicador que revela el estado conexión/desconexión del bluetooth.

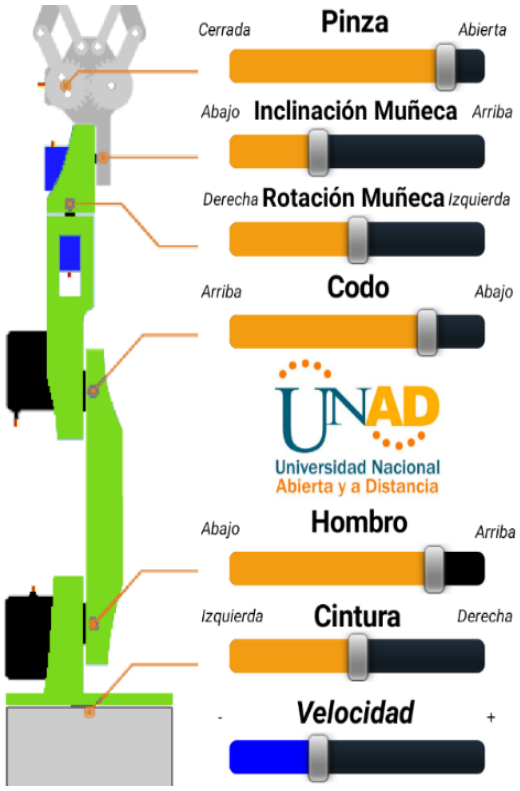


Ilustración 41 funcionamiento Motorola G9 PLUS

Más abajo se encuentran 6 deslizadores, los 5 primeros con valores que representan los movimientos en grados de las articulaciones y que se encuentran previamente definido su posición inicial para que coincida con la del brazo robótico la cual fue llamado estado “Home”, el ultimo representa la velocidad que va de 0 a 60 valores y se posicione inicialmente en 30.

Ilustración 42 Funcionamiento general de la APP

Al final se encuentra el botón GUARDAR, que permite memorizar cada posición indicada por el usuario desde la APP al dar clic, una a una se registran hasta formar el orden algorítmico deseado, es decir, se crea el modo automático de un proceso. Cuando finalice las posiciones se procede a oprimir arrancar, la cual repetirá infinitas veces los movimientos almacenados previamente. También se cuenta con el botón REINICIAR para seguir repitiendo la secuencia o simplemente manipular el robot libremente.

Por último, se cuenta con un conteo de posiciones guardadas en modo automático.

Programación por bloques desde MIT APP Inventor:

Apariencia del componente lógico y programación en bloques.

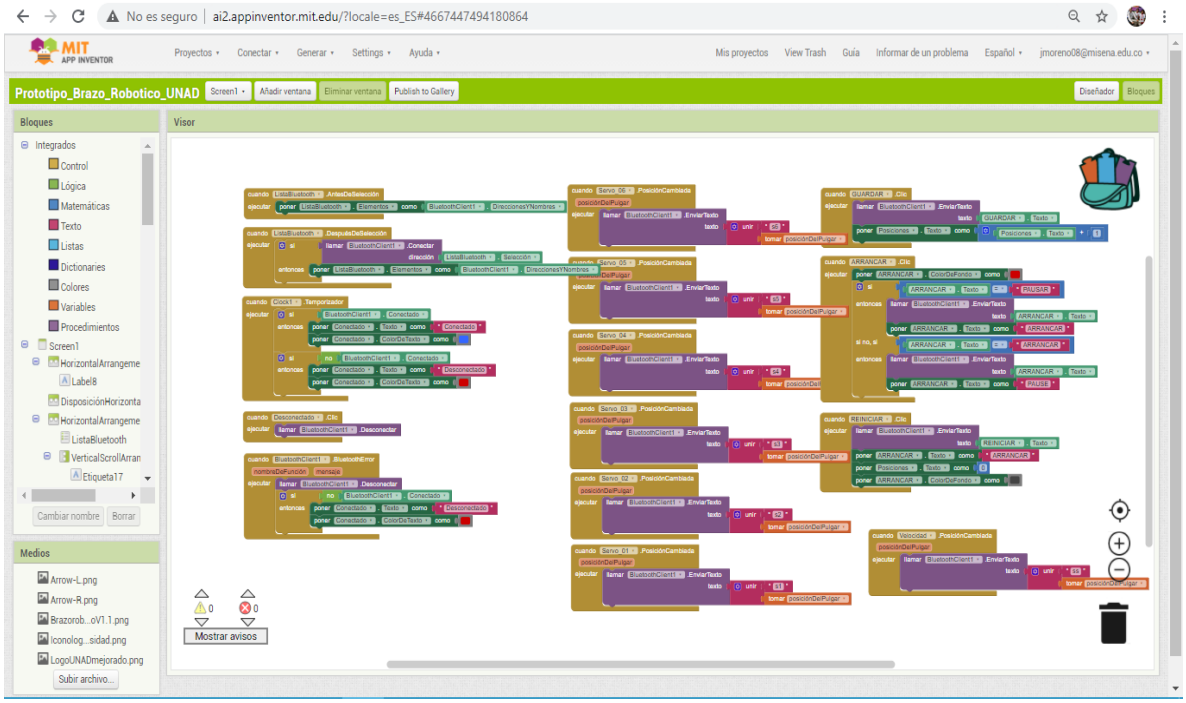
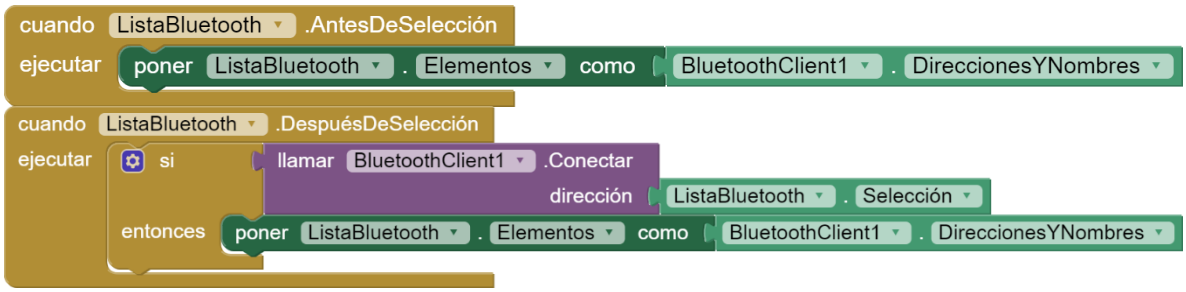
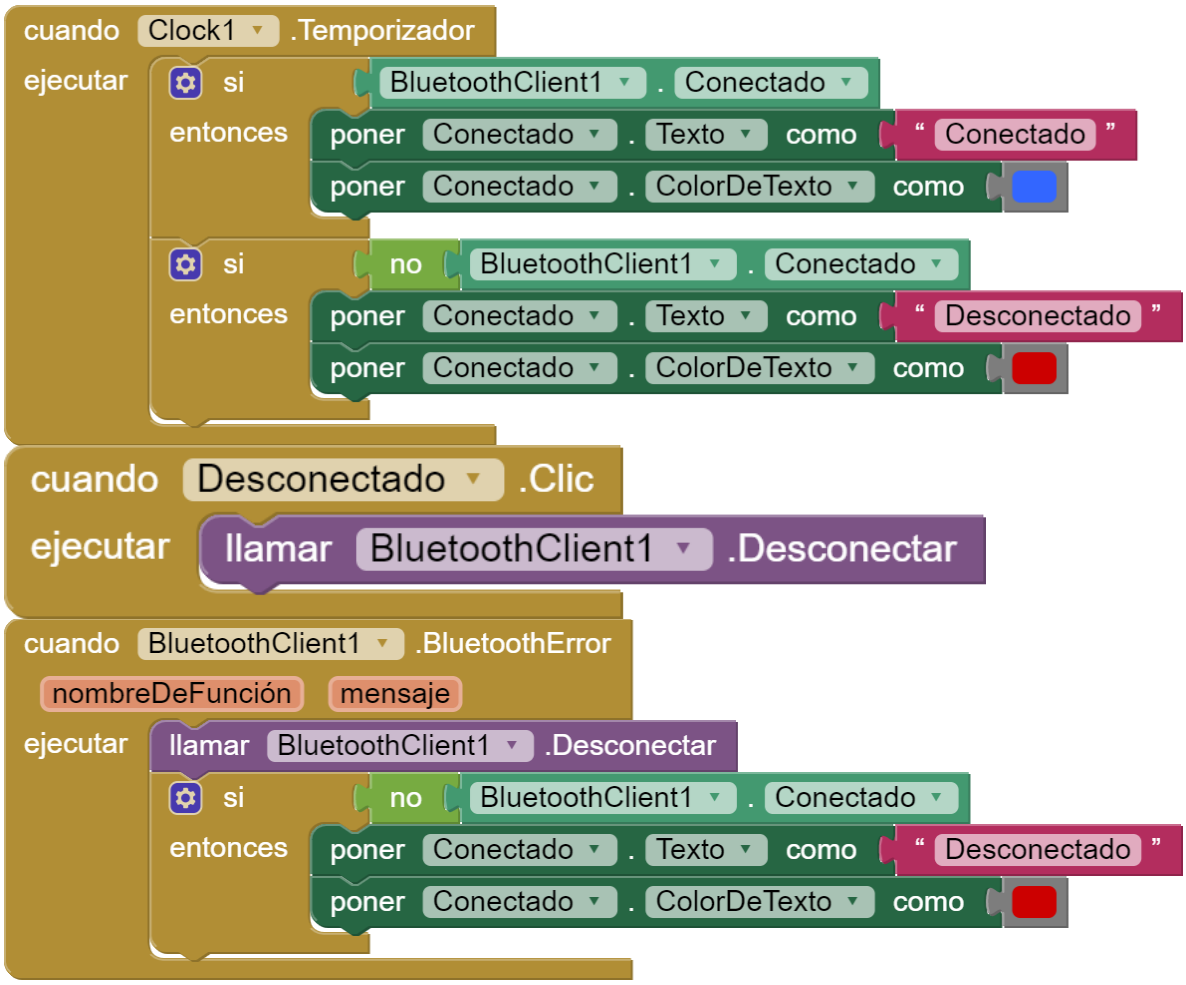


Ilustración 43 MIT APP Inventor

Autoría propia

A continuación, se desglosa cada sección de bloque relacionada:
En esta primera sección se configura las opciones y requisitos para el Bluetooth poder conectarse al Celular:





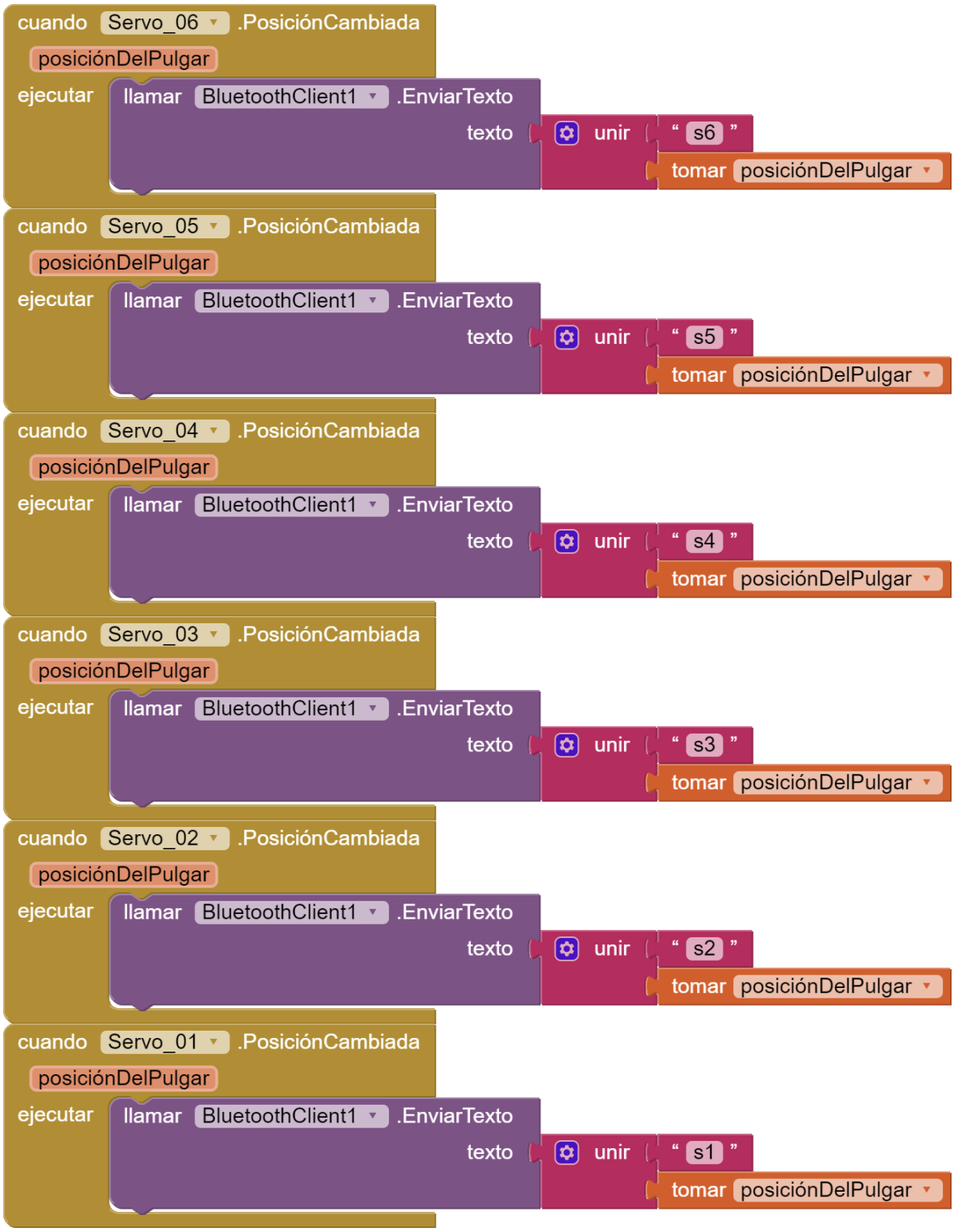
Configuración de bloques APP autoría propia

En esta sección del código de bloques se recibe y envía la nueva posición al Arduino por medio de la función bluetooth ".Enviar Texto", teniendo en cuenta dos cosas, el deslizador que fue variado su valor y el Servomotor que debe controlar. Ejemplo: s210

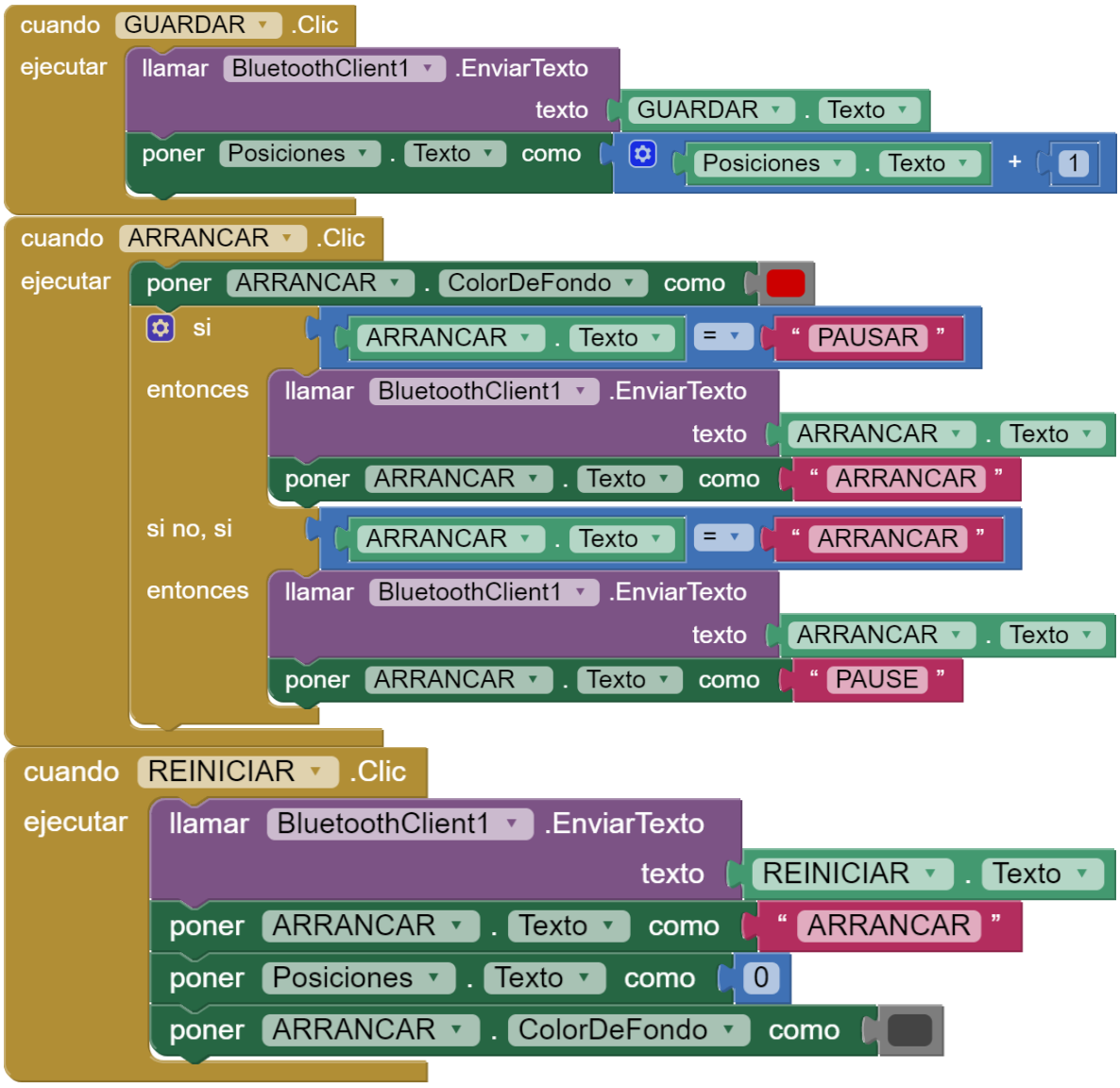
S2: Servomotor 2

10: Valor del deslizador.

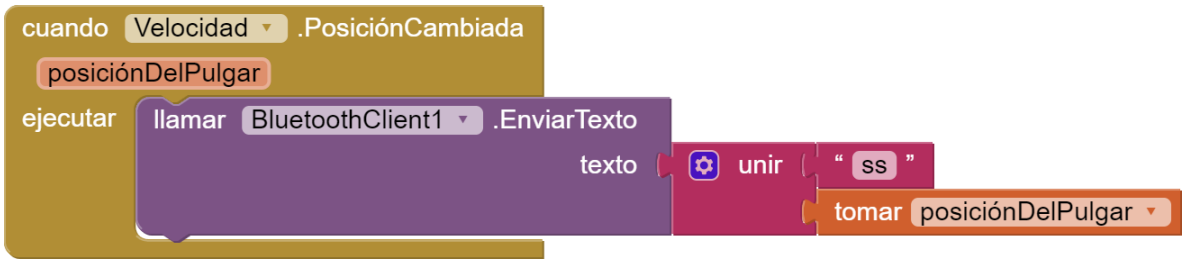
Sistema de bloques autoría propia toda la secuencia


cuando Servo_06 .PosiciónCambiada
posiciónDelPulgar
ejecutar llamar BluetoothClient1 .EnviarTexto
texto unir " s6 "
tomar posiciónDelPulgar
cuando Servo_05 .PosiciónCambiada
posiciónDelPulgar
ejecutar llamar BluetoothClient1 .EnviarTexto
texto unir " s5 "
tomar posiciónDelPulgar
cuando Servo_04 .PosiciónCambiada
posiciónDelPulgar
ejecutar llamar BluetoothClient1 .EnviarTexto
texto unir " s4 "
tomar posiciónDelPulgar
cuando Servo_03 .PosiciónCambiada
posiciónDelPulgar
ejecutar llamar BluetoothClient1 .EnviarTexto
texto unir " s3 "
tomar posiciónDelPulgar
cuando Servo_02 .PosiciónCambiada
posiciónDelPulgar
ejecutar llamar BluetoothClient1 .EnviarTexto
texto unir " s2 "
tomar posiciónDelPulgar
cuando Servo_01 .PosiciónCambiada
posiciónDelPulgar
ejecutar llamar BluetoothClient1 .EnviarTexto
texto unir " s1 "
tomar posiciónDelPulgar

En esta sección se establecen los botones que permiten guardar una serie de instrucciones manuales que posteriormente son ejecutados con la opción arrancar. También da la posibilidad de reiniciar la serie de instrucciones dadas.



Por último, la posibilidad de aumentar el giro por segundo de cada Servomotor gracias a esta última sección del código. La variable “ss” es el deslizador que permite ir desde la velocidad 1 hasta la máxima.



Aplicación con LabVIEW:

Otra de las formas de interacción con el brazo robótico, es a través del software LabView.

LabVIEW (acrónimo de *Laboratory Virtual Instrument Engineering Workbench*) es una plataforma y entorno de desarrollo para diseñar sistemas, con un lenguaje de programación visual gráfico. Recomendado para sistemas hardware y software de pruebas, control y diseño, simulado o real y embebido, pues acelera la productividad. El lenguaje que usa se llama lenguaje G, donde la G simboliza que es lenguaje Gráfico.

Este programa fue creado por *National Instrumets* (1976) para funcionar sobre máquinas MAC, salió al mercado por primera vez en 1986. Los programas desarrollados con LabVIEW se llaman Instrumentos Virtuales, o VIs, y su origen provenía del control de instrumentos, aunque hoy en día se ha expandido ampliamente no sólo al control de todo tipo de electrónica (Instrumentación electrónica) sino también a su programación embebida, comunicaciones, matemáticas, etc. Un lema tradicional de LabVIEW es: "La potencia está en el Software", que con la aparición de los sistemas multinúcleo se ha hecho aún más potente. Entre sus objetivos están el reducir el tiempo de desarrollo de aplicaciones de todo tipo (no sólo en ámbitos de Pruebas, Control y Diseño) y el permitir la entrada a la informática a profesionales de cualquier otro campo. LabVIEW consigue combinarse con todo tipo de software y hardware, tanto del propio fabricante - tarjetas de adquisición de datos, PAC, Visión, instrumentos y otro Hardware- como de otros fabricantes.

En la ilustración 44, se observa el panel de control desarrollado para el brazo, contiene cinco bloques: Indicadores de movimiento, control de movimiento con siete controles deslizantes, de los cuales seis son para controlar el movimiento de las articulaciones y el último para el control de velocidad (PWM), el panel de conexión con el Arduino, el panel para la grabación de posiciones y ejecución. Por último, el panel para establecer los valores del *HOME* del brazo y su ejecución.

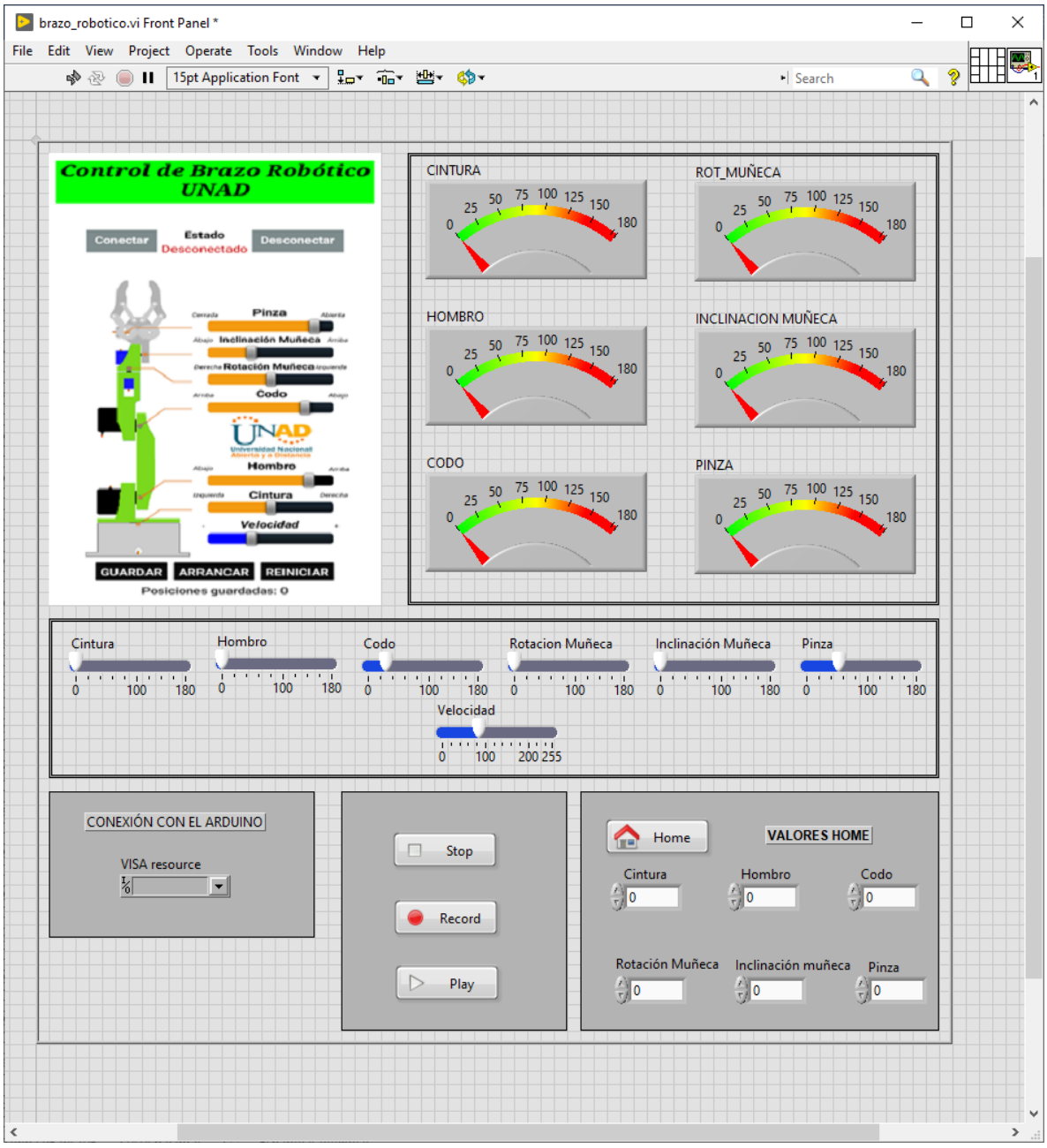


Ilustración 44. Panel de Control del VI del brazo robótico.

El procedimiento es muy simple, se inicia con la carga en el Arduino, del programa LIFA que permite a LabView tomar el control del sistema. Se inicializa la conexión del Arduino y se inicia el control.

Con los controles deslizantes permiten llevar el brazo a una posición deseada y luego dar clic en el botón "*Record*" para almacenarla. En la ilustración 45 se observa la programación del arreglo para esta acción.

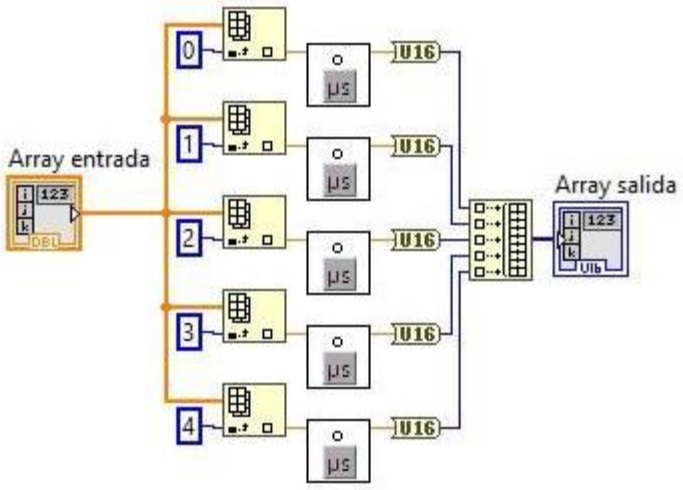


Ilustración 45. Diagrama de bloques del programa Arrays

Para guardar una posición, lo que se requiere hacer es asignar la posición con un número, por lo que esta función dice si el número que se desea utilizar para guardar las coordenadas ya existe. Dicho esto, esta función sirve para mostrar de manera gráfica al usuario las coordenadas ya guardadas en el programa. Si el número que se va a asignar resulta que ya existe, el programa lanzará directamente las coordenadas de dicho punto, avisándonos de que ese punto ya se encuentra asignado.

De igual manera, sirve para recordar qué coordenadas se había asignado para un punto en concreto de este brazo.
A continuación, se muestra una imagen del diagrama de bloques realizado para dicha función.

El inicia con una Array de único punto que va a un Cluster donde está dicho punto más todas las partes del brazo y un posible comentario.

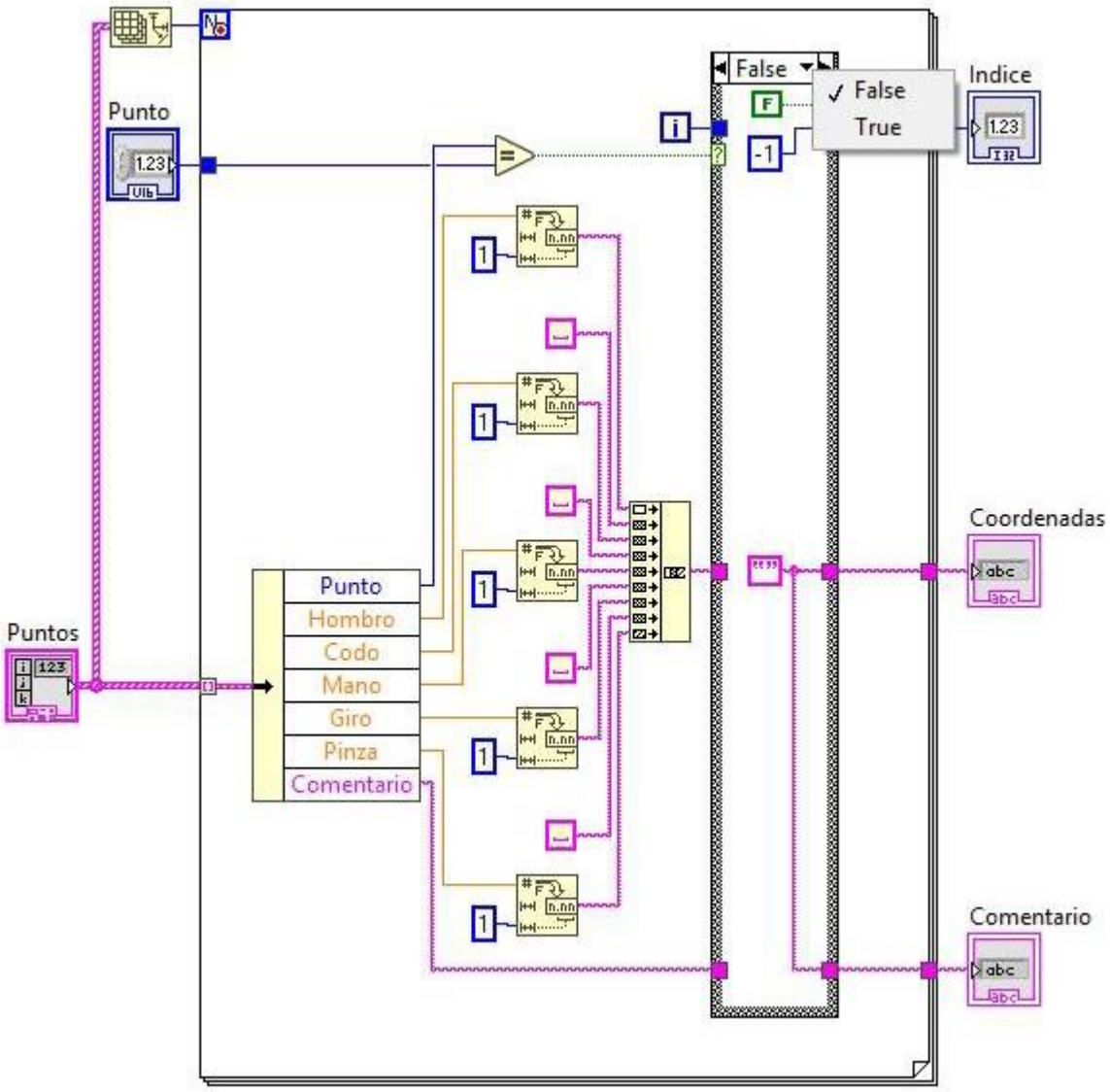


Ilustración 46. Bloque de programa para la búsqueda de un punto

El programa guarda las posiciones en memoria para poder llamarlo posteriormente. Guarda cada punto con una precisión de tres decimales, aunque como está en la función de “Buscar punto”, los muestra únicamente con uno solo decimal. También es posible crear fichero, editar y abrir. Además, permite hacerlo en modo “sólo lectura” ayudando a proteger el archivo.

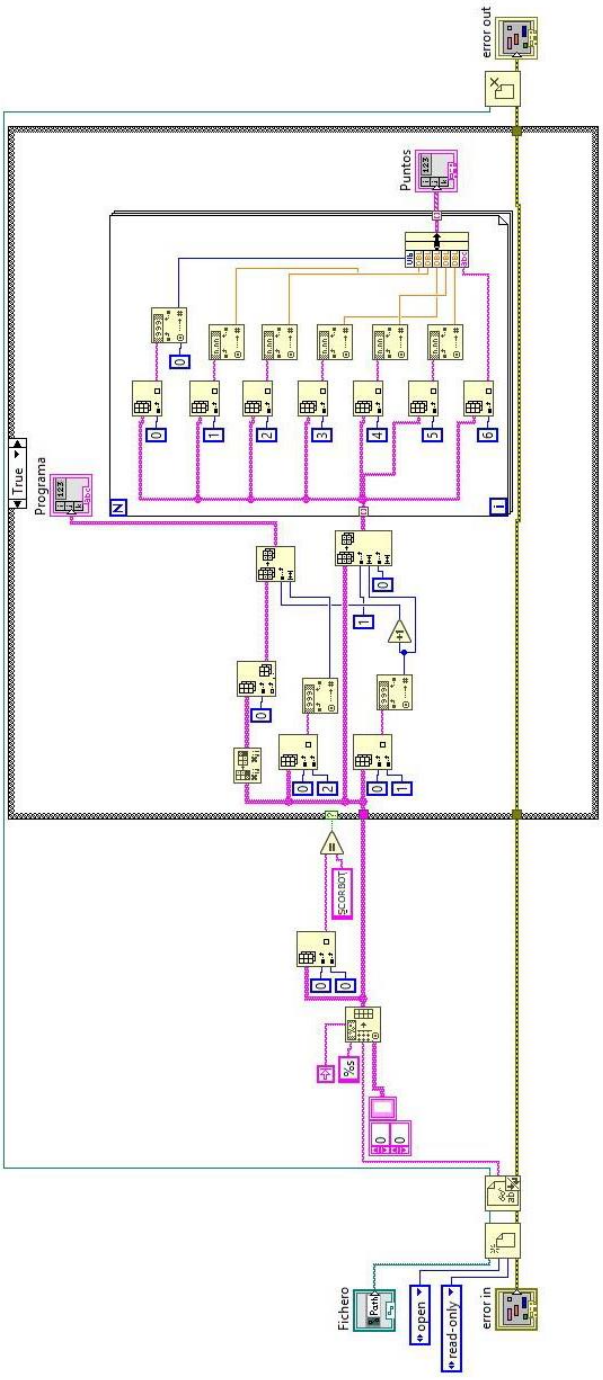


Ilustración 47. Diagrama de bloques de la función guardar

Y para ejecutar los movimientos guardados se tiene la función leer:

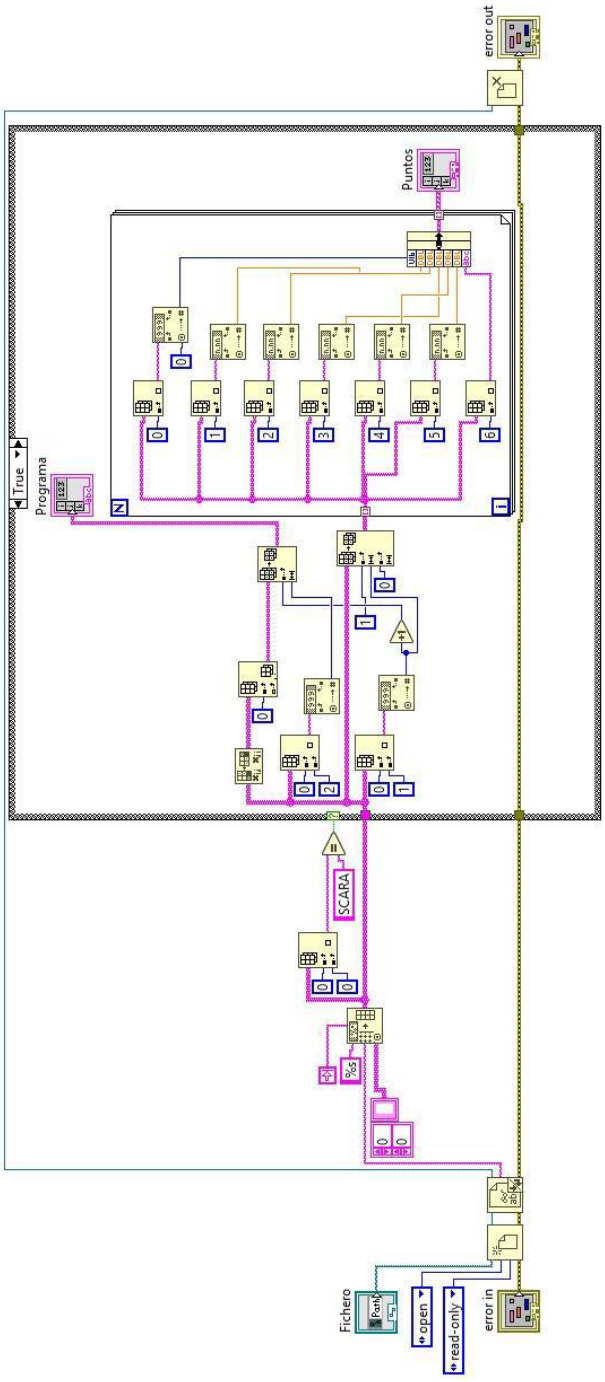


Ilustración 48. Diagramas de bloques de la función leer

La función para obtener un punto consiste en poder constatar si un punto ya existe, por lo que facilita a no sobrescribir. Lo cual es muy útil a la hora de poder recordar cual es la posición de puntos anteriormente guardados.

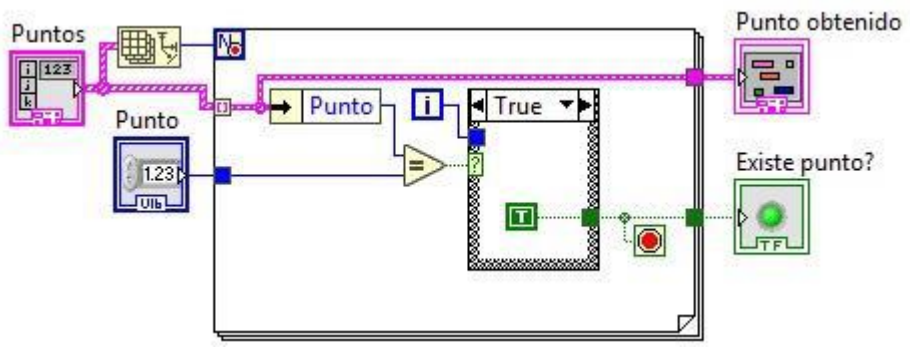


Ilustración 49. Diagrama de bloques de la función para registrar un punto.

Otras funciones creadas son las que permiten almacenar los datos de los puntos en la memoria del computador o para leerlos desde el computador. Estas funciones son Array-String y String-Array. La primera hace la conversión de números a una cadena de caracteres y la segunda la función inversa.

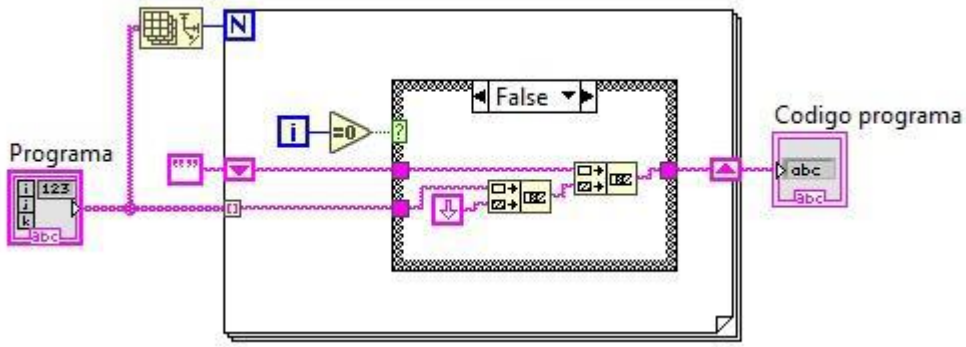
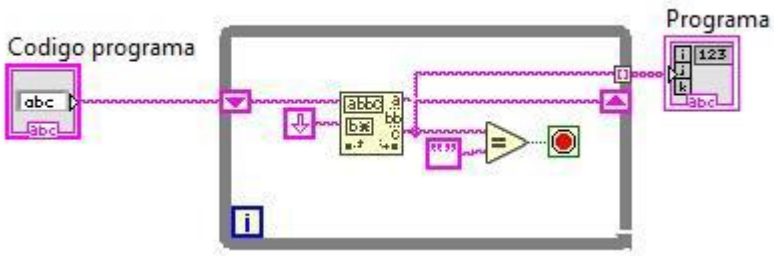


Ilustración 50. Diagrama de bloques de la función Array-String



```
SCORBOT  8        16
1        24,000   25,000   70,000   80,000   100,000 P1 arriba pinza abierta
2        24,000   25,000   10,000   80,000   100,000 P1 posición abajo pinza abierta
3        24,000   25,000   10,000   80,000   160,000 P1 posición abajo pinza cerrada
4        24,000   25,000   120,000  80,000   160,000 P1 posición arriba pinza cerrada
5        129,000  85,000   120,000  80,000   160,000 P1 posición final arriba pinza cerrada
6        129,000  85,000   0,000    80,000   160,000 P1 posición final abajo pinza cerrada
7        129,000  85,000   0,000    80,000   100,000 P1 posición final abajo pinza abierta
8        129,000  85,000   80,000   80,000   100,000 P1 posición final arriba pinza abierta
MOVER 1
PAUSA 1000
MOVER 2
PAUSA 1000
MOVER 3
PAUSA 1000
MOVER 4
PAUSA 1000
MOVER 5
PAUSA 1000
MOVER 6
PAUSA 1000
MOVER 7
PAUSA 1000
MOVER 8
INICIO
```

Ilustración 51. Formato del archivo para ejecutar.

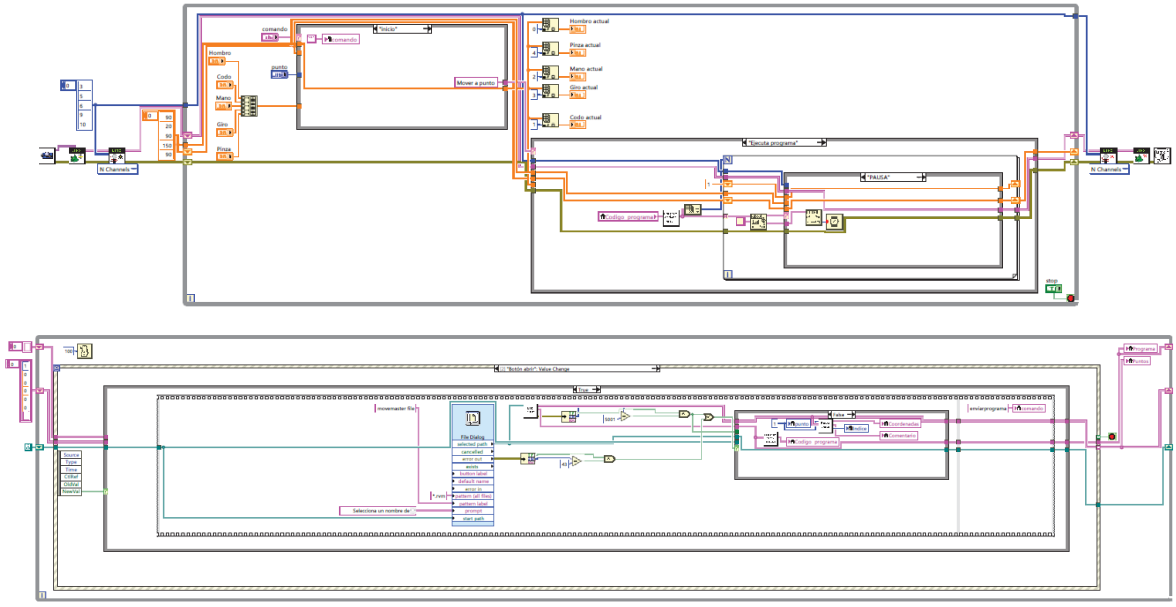


Ilustración 52. Diagrama de bloques sin desglosar.

CONCLUSIONES

Como resultado final se obtuvo el brazo robótico funcional, con todas sus características, con pequeños ruidos en la señal debido a la protoboard que genera estas corrientes de fuga o ruidos, por lo cual se acudió a borneras y a soldar directamente los cables de los servomotores para evitar empalmes innecesarios y perjuicioso obtenidos previamente.

La conexión entre el dispositivo móvil, el Arduino y el brazo es una conexión bastante rápida, de acción inmediata donde los valores de inicio o limitantes son importantes para el brazo, no necesariamente las articulaciones deben tener los 180 grados de movilidad que presenta el servo, ya que algunas de estas articulaciones no deben alcanzar algunos topes porque pueden sufrir daños físicos por choque entre eslabones.

La alimentación resulta ser fundamental, ya que tener activos seis servos requiere de una buena entrada de voltaje, lo que generaba problemas al inicio de las pruebas del brazo, ya que la fuente con la que se hicieron las pruebas no superaba el amperio de corriente, lo cual generaba un error a la hora de moverlo y utilizar la aplicación móvil, esta mostraba “error de tubería rota".

La utilización de la comunicación serial del Arduino por sus pines normales de rx y tx se debe de realizar con precaución, ya que el programa no se puede cargar al Arduino si estos pines se encuentran ocupados, por esto se toma la decisión de establecer comunicación por pines diferentes.

La programación en bloques a la hora de construir los procesos del aplicativo es sencilla debido a la facilidad de entendimiento, precisamente gracias a los bloques que hacen todo más visual, más entendible a la hora de abrir funciones y subprocesos, los cuales los estudiantes interesados no tendrán problemas a la hora de realizar este paso o algunas modificaciones.

El bajo costo de la construcción de esta herramienta es importante, debido a que más estudiantes tendrán la facilidad de construir este modelo; así mismo mejorando sus conocimientos y habilidades con materias a fines.

Se toman las referencias de los materiales utilizados ya que prestan servicios muy buenos para este proyecto a muy bajo costo, cumplen con los objetivos del

proyecto. Además, su fácil accionamiento, programación y utilización hace que más personas se acerquen.

Extrapolando los valores, funciones, acciones, características de este proyecto, se puede llegar a la construcción de demás dispositivos afines y similares. Lo que hace que el estudiante lleve sus conocimientos adquiridos a prueba para satisfacer en el campo industrial, medico, comercial, entre otras, las necesidades existentes.

Enlace del video del prototipo en modo manual y en modo automático:

M. Manual: https://drive.google.com/file/d/1bvSbOECIf6vTfWDjuqTZoGiO55pGaFBq/view?usp=sharing
M. Automático: https://drive.google.com/file/d/1Z8xYYoIfleV4-hKivfN7hlj4yxBqTcct/view?usp=sharing

Como resultado se obtuvo este prototipo funcional:

Imagen propia

BIBLIOGRAFÍA

ASOCIACIÓN COLOMBIANA DE FACULTADES DE INGENIERÍA -ACOFI. Actualización y Modernización curricular en Ingeniería de Sistemas. ACOFI-ICFES. Bogotá, 1996.

BLANCHARD SEAVER, Benjamin. Administración de la Ingeniería de Sistemas.1 ed. Grupo Noriega Editores, Alfaomega Rama, 1993.

BOXALL, John. Arduino workshop: A Hands-on Introduction with 65 Projects. 1 ed. San Francisco, California, 2012.

Disponible en: http://bibliotecavirtual.unad.edu.co/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=edsgao&AN=edsgcl.337720694&lang=es&site=eds-live&scope=site

WILSON, Brian Sistemas: ¡Conceptos, Metodología y Aplicaciones! Ed. Megabyte. 1993

CHECKLAND, Peter. Pensamiento de Sistemas, práctica de sistemas. México D.F, Limusa Noriega Editores, 2001.

ABARCA JIMÉNEZ, Griselda Stephany y MARES CARREÑO, Jesús y CORONA RAMÍREZ, Leonel Germán. Sensores y Actuadores Aplicaciones con Arduino. México D. F, Instituto Politécnico Nacional, 2014. 39-108 p.
Disponible en: http://bibliotecavirtual.unad.edu.co:2460/lib/unadsp/reader.action?ppg=2&docID=4569609&tm=1527546697645

FAJARDO, Carlos. Primeros pasos con el IDE de Arduino, Colombia, Universidad Nacional Abierta y a Distancia, 2016. video
Disponible en: http://hdl.handle.net/10596/9831

MILANÉS HERMOSILLA, Daily, y CASTILLA PÉREZ, Alejandro. Generación de trayectorias para el brazo robótico (ArmX). Ingeniería Electrónica, Automática y Comunicaciones, 2019. 58-71 p.
Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59282016000300006&lng=es&tlng=es.

GRECH MAYOR, Pablo. Introducción a la ingeniería. Un enfoque a través del diseño. 2 ed. Pearson Educación de Colombia. Cali,Colombia, Pontificia Universidad Javeriana, 2013. 88-184-318-362-396 p

INTELITEK, Manual del usuario del robot Scorbot ER 9 Pro, Estados unidos, 2011 Disponible en: http://www.intelitekdownloads.com/Manuals/Robotics/Spanish/Scorbot-ER-9Pro-ES-B.pdf

SPONG, Mark y Hutchinson, Seth y VIDYASAGAR, Mathukumalli. Robot Modeling and Control. John Wiley & Sons, Hoboken, NJ. 1 ed. USA, 2006.

MUHAMMAD, Saleheen Aftab y MUHAMMAD, Shafiq. Manipulador robótico TQ MA3000 5 DOF. Universidad Sultan Qab, diciembre de 2014
Disponible en: https://www.researchgate.net/figure/TQ-MA3000-5-DOF-Robotic-Manipulator_fig1_272163214

NOERGAARD, Tammy. Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers. Waltham, MA, Estados Unidos, 2005. Chapter 15-16 p.
Disponible en: http://bibliotecavirtual.unad.edu.co/login?url=https://bibliotecavirtual.unad.edu.co:2969/login.aspx?direct=true&db=e000xww&AN=195129&lang=es&site=ehost-live&ebv=EB&ppid=pp_5

NOGUERA TORRES, Ariana Pilar. Generalidades de Sistemas Embebidos. Universidad nacional abierta y a distancia. Colombia, 2018. Disponible en: http://hdl.handle.net/10596/22789

RUBIANO LLORENTE, Jaime y BOLIVAR MARÍN, Fabián y TORRES SILVA, Pedro. Diseño y desarrollo de una unidad de control para el Robot Educativo SCORBOT ER 9 Pro basada en sistemas de procesamiento de bajo costo. Universidad Nacional Abierta y a Distancia, Neiva, Colombia, 2018.

NIKU, Saeed Benjamin. Niku. Introduction to Robotics: Analysis, Systems, Applications. Prentice Hall, Upper Saddle River, NJ, WRIGHT. 1 ed. California, USA, 2001.g

RESUMEN ANALITICO EDUCATIVO RAE

Título del texto	DISEÑO DE LAS UNIDADES DE CONTROL PARA CADA ARTICULACIÓN DEL ROBOT EDUCATIVO PROTOTIPO SIMILAR A SCORBOT ER 9 PRO BASADA EN SISTEMAS DE PROCESAMIENTO DE BAJO COSTO
Nombres y Apellidos del Autor	LEONARDO ANDRES DÍAZ BARRETO JEFERSON MORENO CABRERA DIEGO MAURICIO PARRA ALMARIO
Año de la publicación	**2020**

Resumen del texto:

Actualmente se necesita tener nuevas proyecciones educativas para fomentar la innovación y la creatividad de los procesos académicos, es por eso por lo cual la universidad nacional abierta y a distancia fomenta la vinculación a prácticas profesionales adecuadas para que el estudiante interactúe en un medio adecuado dirigido a la solución de problemas, esto se proyecta que se diseñe un prototipo en el cual se de hardware y software necesarios para el controlador de cada articulación del robot Scorbot ER-9 Pro.

Esto se da basado en un prototipo similar para poder implementar este sistema de tal manera que se pueda basar en el robot Scorbot ER-9 Pro para aplicación de controladores, mediante creación de piezas en 3d manejo de control remoto y adecuación de sensores de posicionamiento.

Todo basado en la investigación de lo que se viene desarrollando en el proyecto de PIE_G_29_18ECBTI, que están adelantando los Ingenieros Fabian Bolívar, Jaime Rubiano y Pedro Torres, del CCAV de Neiva, tiene como objeto: “Diseño de las unidades de control para cada articulación del Robot Educativo PROTOTIPO SIMILAR a SCORBOT ER 9 Pro basada en sistemas de procesamiento de bajo costo”.

Palabras Claves	Control abierto, control cerrado, cinemática directa e inversa, robots, tarjetas de control, Arduino, Raspberry, comunicación I2C, código C++, sistemas SCADA, LabVIEW

Problema que aborda el texto:
Proyecto aplicado el propiciar nuevos aportes e investigaciones en sistemas de control embebido y con unas restricciones claramente conocidas. Bajo este argumento, la propuesta se encamina a diseñar a partir de los principios

cinemáticos directos como inverso, los controles para las articulaciones del brazo robótico. Así poder dar una proyección a el manejo de un brazo robótico como aplicación a nuevas tecnologías.
Objetivos del texto: Diseñar y desarrollar las unidades de control para cada articulación del Robot Educativo PROTOTIPO SIMILAR a SCORBOT ER 9 Pro basada en sistemas de procesamiento de bajo costo
Hipótesis planteada por el autor: Como se debería tener aplicación de nuevas tecnologías a los problemas aplicativos en la proyección de impulsar a la investigación en la actualidad
Tesis principal del autor: Teniendo en cuenta la actualidad de la evolución de las tecnologías, se puede relacionar los sistemas embebidos con el internet de las cosas de tal manera de que los sistemas embebidos son sistemas computacionales por así llamarlos en los cuales se enfoca a dar soluciones en tiempo real a muchas tareas y debido al avance se tienen que conectar a Internet para enfocarse en la relación del sistema y el avances tecnológico actual de acuerdo a que los dispositivos y los aparatos electrónicos están evolucionando a velocidades extremas dando la facilidad a las personas de acceder a circuitos integrados y componentes electrónicos en una amplia gama y a grandes distancias en tiempo real.
Argumentos expuestos por el autor: La participación en la implementación de un sistema de control modular que permita a los estudiantes tener acceso a cada una de las articulaciones de forma independiente dentro de todo el sistema. En consecuencia, los estudiantes podrán no solo conocer y familiarizarse con el modelo de funcionamiento de cada articulación, sino también con los principios de funcionamiento de un brazo robótico tipo industrial
Conclusiones del texto: • Aplicación de tecnologías La programación secuencial no se adapta a las necesidades de control derivadas de la interacción del robot con un mundo inherentemente asíncrono. Abstracción del hardware específico del robot: servos, encoders etc. Manejo de la complejidad en operaciones, comunicación y distribución de datos. • Proyección en la investigación Programación de robot mediante la ejecución de software customizado, haciendo uso de librerías para la lectura en puerto serie y gestión de las interfaces de propósito general. Programación de software bajo el Sistema Operativo ROS, integración mediante librerías dedicadas de plataformas de control de propósito general tales como Arduino y Raspberry Pi.
Bibliografía citada por el autor:

ASOCIACIÓN COLOMBIANA DE FACULTADES DE INGENIERÍA -ACOFI. Actualización y Modernización curricular en Ingeniería de Sistemas. ACOFI-ICFES. Bogotá, 1996.

BLANCHARD SEAVER, Benjamin. Administración de la Ingeniería de Sistemas. 1 ed. Grupo Noriega Editores, Alfaomega Rama, 1993.

BOXALL, John. Arduino workshop: A Hands-on Introduction with 65 Projects. 1 ed. San Francisco, California, 2012.
Disponible en: http://bibliotecavirtual.unad.edu.co/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=edsgao&AN=edsgcl.337720694&lang=es&site=eds-live&scope=site

WILSON, Brian. Sistemas: ¡Conceptos, Metodología y Aplicaciones! Ed. Megabyte. 1993.

CHECKLAND, Peter. Pensamiento de Sistemas, práctica de sistemas. México D.F, Limusa Noriega Editores, 2001.

ABARCA JIMÉNEZ, Griselda Stephany y MARES CARREÑO, Jesús y CORONA RAMÍREZ, Leonel Germán. Sensores y Actuadores Aplicaciones con Arduino. México D. F, Instituto Politécnico Nacional, 2014. 39-108 p.
Disponible en: http://bibliotecavirtual.unad.edu.co:2460/lib/unadsp/reader.action?ppg=2&docID=4569609&tm=1527546697645

FAJARDO, Carlos. Primeros pasos con el IDE de Arduino, Colombia, Universidad Nacional Abierta y a Distancia, 2016. video
Disponible en: http://hdl.handle.net/10596/9831

MILANÉS HERMOSILLA, Daily, y CASTILLA PÉREZ, Alejandro. Generación de trayectorias para el brazo robótico (ArmX). Ingeniería Electrónica, Automática y Comunicaciones, 2019. 58-71 p.
Disponible en: http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1815-59282016000300006&lng=es&tlng=es.

GRECH MAYOR, Pablo. Introducción a la ingeniería. Un enfoque a través del diseño. 2 ed. Pearson Educación de Colombia. Cali,Colombia, Pontificia Universidad Javeriana, 2013. 88-184-318-362-396 p

INTELITEK, Manual del usuario del robot Scorbot ER 9 Pro, Estados unidos, 2011 Disponible en: http://www.intelitekdownloads.com/Manuals/Robotics/Spanish/Scorbot-ER-9Pro-ES-B.pdf

SPONG, Mark y Hutchinson, Seth y VIDYASAGAR, Mathukumalli. Robot Modeling and Control. John Wiley & Sons, Hoboken, NJ. 1 ed. USA, 2006.

MUHAMMAD, Saleheen Aftab y MUHAMMAD, Shafiq. Manipulador robótico TQ MA3000 5 DOF. Universidad Sultan Qab, diciembre de 2014
Disponible en: https://www.researchgate.net/figure/TQ-MA3000-5-DOF-Robotic-Manipulator_fig1_272163214

NOERGAARD, Tammy. Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers. Waltham, MA, Estados Unidos, 2005. Chapter 15-16 p.
Disponible en: http://bibliotecavirtual.unad.edu.co/login?url=https://bibliotecavirtual.unad.edu.co:2969/login.aspx?direct=true&db=e000xww&AN=195129&lang=es&site=ehost-live&ebv=EB&ppid=pp_5

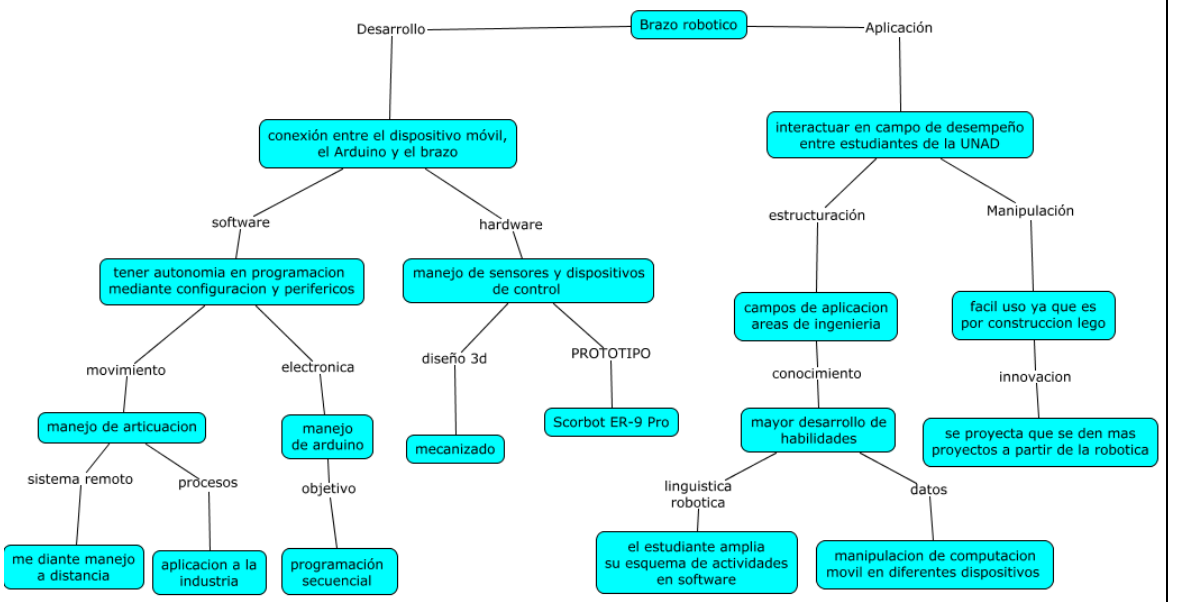
NOGUERA TORRES, Ariana Pilar. Generalidades de Sistemas Embebidos. Universidad nacional abierta y a distancia. Colombia, 2018. Disponible en: http://hdl.handle.net/10596/22789

RUBIANO LLORENTE, Jaime y BOLIVAR MARÍN, Fabián y TORRES SILVA, Pedro. Diseño y desarrollo de una unidad de control para el Robot Educativo SCORBOT ER 9 Pro basada en sistemas de procesamiento de bajo costo. Universidad Nacional Abierta y a Distancia, Neiva, Colombia, 2018.

NIKU, Saeed Benjamin. Niku. Introduction to Robotics: Analysis, Systems, Applications. Prentice Hall, Upper Saddle River, NJ, WRIGHT. 1 ed. California, USA, 2001.g

Nombres y apellidos de quienes elaborarón este RAE	LEONARDO ANDRES DÍAZ BARRETO JEFERSON MORENO CABRERA DIEGO MAURICIO PARRA ALMARIO
Fecha en que se elaboró este RAE	Noviembre de 2020

Imagen que resume e interconecta los principales conceptos encontrados en el texto:



Comentarios finales

Como resultado final está el brazo funcional, con todas sus características, con pequeños ruidos en la señal debido a la protoboard que genera estas corrientes de fuga o ruidos, lo anterior corregido con unas placas usadas como borneras y los cables de los motores sin empalmes para un mejor desempeño.

La conexión entre el dispositivo móvil, el Arduino y el brazo es una conexión bastante rápida, de acción inmediata donde los valores de inicio o limitantes son importantes para el brazo, no necesariamente las articulaciones deben tener los 180 grados de movilidad que presenta el servo, ya que algunas de estas articulaciones no precisan de todos estos grados.