Proyecto Final

Introducción

Hoy en día los brazos robóticos se utilizan a menudo para realizar tareas complejas o muy repetitivas, con el fin de eliminar el tipo de errores que una persona cansada cometería después de llevar a cabo dicha tarea tediosa por largos periodos de tiempo.

Los avances tecnológicos han permitido que dispositivos, con una amplia gama de sensores, sean ideados y que, indudablemente, se encuentren en aplicaciones de diversos ámbitos para el entorno.

Con base en lo anterior, podemos afirmar que existe un interés común por realizar los cálculos necesarios que nos lleven a fabricar un modelo funcional de un brazo robótico.

Objetivos

Diseñar y prototipar un brazo robótico con cuatro grados de libertad y un efector final.

Realizar los análisis de los elementos de dicho brazo robótico para caracterizar el diseño por medio de sensores, actuadores y un material previamente seleccionado.

Obtener el modelado cinemático directo de cada uno de los eslabones que conforma nuestro modelo.

Fabricar, en el mundo real, el sistema diseñado y cumplir con un conjunto de pruebas básicas para su correcto funcionamiento.

Planteamiento del Problema

Se presenta la relevancia y contextualización de la cinemática de un sistema de brazo robótico con matriz de Denavit-Hartenberg, la cual permite diseñar de manera concreta el movimiento que se desea en un sistema cinemático. Se comprenden tanto los movimientos de rotación y traslación, como los movimientos angulares respectivos de cada elemento.

El presente proyecto planteado a través de la relación de las matrices con el fenómeno físico del movimiento tiene como premisa resolver un conjunto de circulación dentro del sistema robótico para controlar de manera eficiente el modelo físico del brazo adecuado para cinco grados de libertad.

Materiales

Para la parte electrónica de nuestro brazo robótico tenemos pensado la siguiente lista con el fin de cubrir el funcionamiento del brazo y un sistema con el cual podremos controlarlo.

- Arduino Uno o Arduino Mega 2560 (1).
- Servomotor metálico TowerPro MG90S (4).
- Micro Servomotor metálico TowerPro MG90 (2).
- Micro Switch Push Button 4 Pines (8).

- Interruptor de 5V (1).
- Resistencia de 10 KΩ (8).
- Resistencia de 220 Ω (1).
- Led verde (1).
- Protoboard 1 bloque, 2 tiras, 400 puntos (1).
- Cargador de Carga Rápida 5V, hasta 2A. (1)
- Cable USB-MicroUSB (1).

En cuestión de la parte del cuerpo se contempla el uso de MDF, cartoncillo o grabar las piezas mediante impresión 3D, dependiendo del costo.

Acerca del servomotor y cálculo de potencias, enlistamos las variables a tomar en cuenta:

- Peso: 13.4 g
- Dimensiones: 22.5 x 12 x 35.5 cm
- Torque: 1.8 kgf·cm (4.8V)
- Velocidad de operación: 0.1 s/60 degree
- Voltaje de operación: 4.8 V
- Ancho de banda: 5 µs

Marco teórico

Un robot se define como un dispositivo mecánico que posee articulaciones móviles enfocadas en la manipulación, cuya función es desempeñar actividades bajo la supervisión humana directa o con software previamente definido.

Robots Manipuladores

Un robot articulado es un manipulador reprogramable multifuncional y es diseñado con el fin de mover materiales, piezas o dispositivos especializados a través de movimientos programados variables para la realización de una diversidad de tareas específicas. A su vez, se deben definir los grados de libertad del manipulador, definidos como el tipo de movimientos independientes que el robot tiene la posibilidad de realizar.

Otro aspecto importante para considerar es la capacidad de carga que tiene el robot a la hora de manipular objetos. Lo anterior, en términos técnicos, se describe como los pares necesarios para mover la carga, cuyo valor varía según la configuración que el robot posee, esta carga manipulable por todo el volumen manipulado recibe el nombre de “Carga Útil”.

Por último, se debe definir el volumen de trabajo del manipulador, el cual es conocido como aquel espacio dentro del que se puede desplazar el punto en el que se ensamble el efector final.

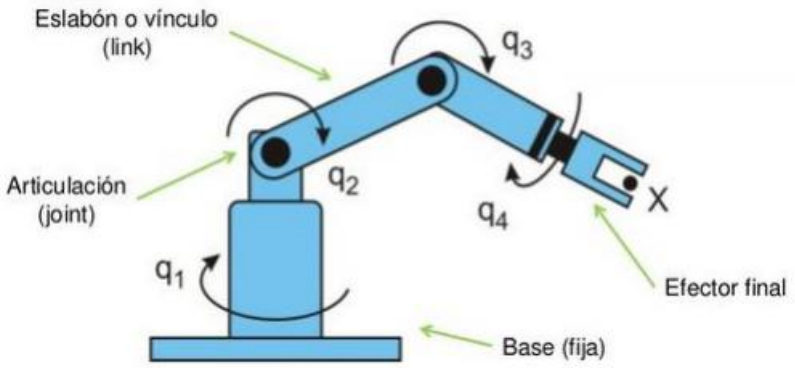


Figura 1.0 “Estructura básica de un manipulador”

Configuración de los robots manipuladores

La configuración de un robot manipulador se define a partir del tipo de articulación que éste tenga, sin embargo, la más usada es la articulación de rotación debido a que suministra solo un grado de rotación consistente alrededor de un eje de articulación. En nuestro caso utilizaremos este último tipo de articulaciones, debido a su sencillez en el cálculo.

Tinkercad

Como base de desarrollo de proyecto se hizo uso de la plataforma Tinkercad que es un software de uso gratuito en línea que permite el diseño, implementación y simulación de sistemas integrados con Arduino. La gran ventaja de esta plataforma es que la simulación se puede llevar a cabo mediante la conexión de hardware y la subida de un código al sistema de Tinkercad. La interfaz de uso es bastante sencilla ya que se pueden tener sistemas complejos y la sencillez de la plataforma sigue siendo fácil de usar. Inclusive la comunicación serial está habilitada para entrada/salida por consola. Para el desarrollo del presente proyecto se hace uso de los siguientes elementos existentes en Tinkercad:

- LCD 16x2
- Arduino Uno
- Servomotores
- Potenciómetros
- PushBotton
- Switch
- Resistencias

Sistema de Control

Un sistema de control es un conjunto de dispositivos de distinto orden que pueden ser del tipo eléctrico, hidráulico, neumático y mecánico cuya interacción tiene el objetivo de manejar un sistema y reducir los errores (de cualquier tipo). Las partes fundamentales de un sistema de control se pueden expresara como una variable a controlar (posición para nuestra aplicación), un actuador (servomotores) y un punto de referencia que hace la comparación de la variable a controlar.

Los tipos de sistemas de control se clasifican en dos grupos: de lazo abierto o de lazo cerrado. Los de lazo abierto solo constan con los 3 componentes básicos que se mencionaron con anterioridad, la desventaja de estos sistemas es que su rango de error es muy amplio y solo pueden ser usados en aplicaciones que

no requieran demasiada precisión. Los sistemas de control de lazo cerrado en cambio tienen un elemento de retroalimentación que en el punto de suma compara la entrada deseada con la lectura de la retroalimentación que es dada por un sensor. El rango de error será muchísimo menor ya que cuentan con un sistema de retroalimentación.

Componentes

- LCD: Liquid Crystal Display es un dispositivo electrónico que muestra de manera gráfica cualquier información que se le envíe desde caracteres hasta valores de distintos sensores. El tamaño de la pantalla varía de acuerdo con el modelo, para este caso se hace uso de una de 16x2.

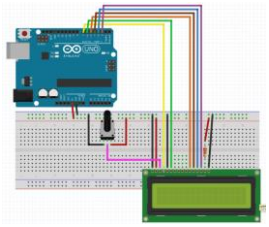


Ilustración 1 Conexión LCD

- Servomotor: Los servomotores nos permiten mantener una posición indicada durante un tiempo indicado y más importante aún nos da la posibilidad de controlar a nuestro gusto la velocidad con la que se llega a un ángulo especifico de posicionamiento. Es por esto por lo que el control de posición y accionamiento se realizará por medio de servomotores. Cada servomotor estará instalado en una articulación. Para controlar las articulaciones en conjunto se hace uso del microprocesador Arduino que nos facilitará el control de todo el sistema.

  El método de control para los servomotores será por medio de señales PWM generadas por el microcontrolador Arduino. El pulso de trabajo varía entre 1ms y 2 ms con un periodo de Ms (50 Hz) lo cual quiere decir que la velocidad máxima de cambio será cada 20 ms (esto depende del tipo de servo).

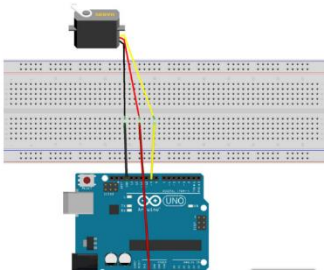


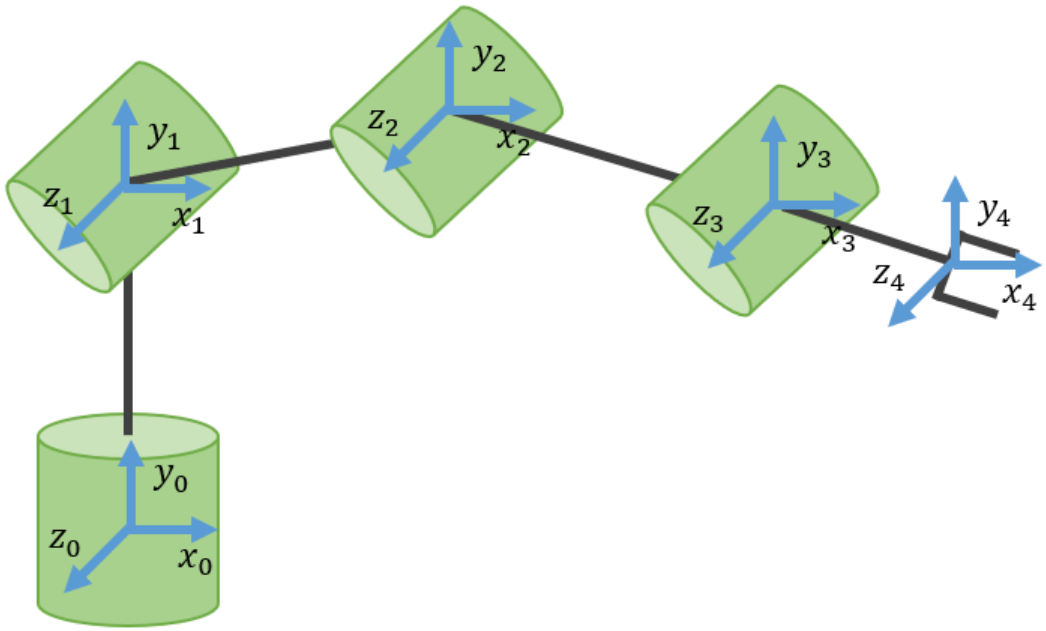
Ilustración 2 Conexión de Servomotor

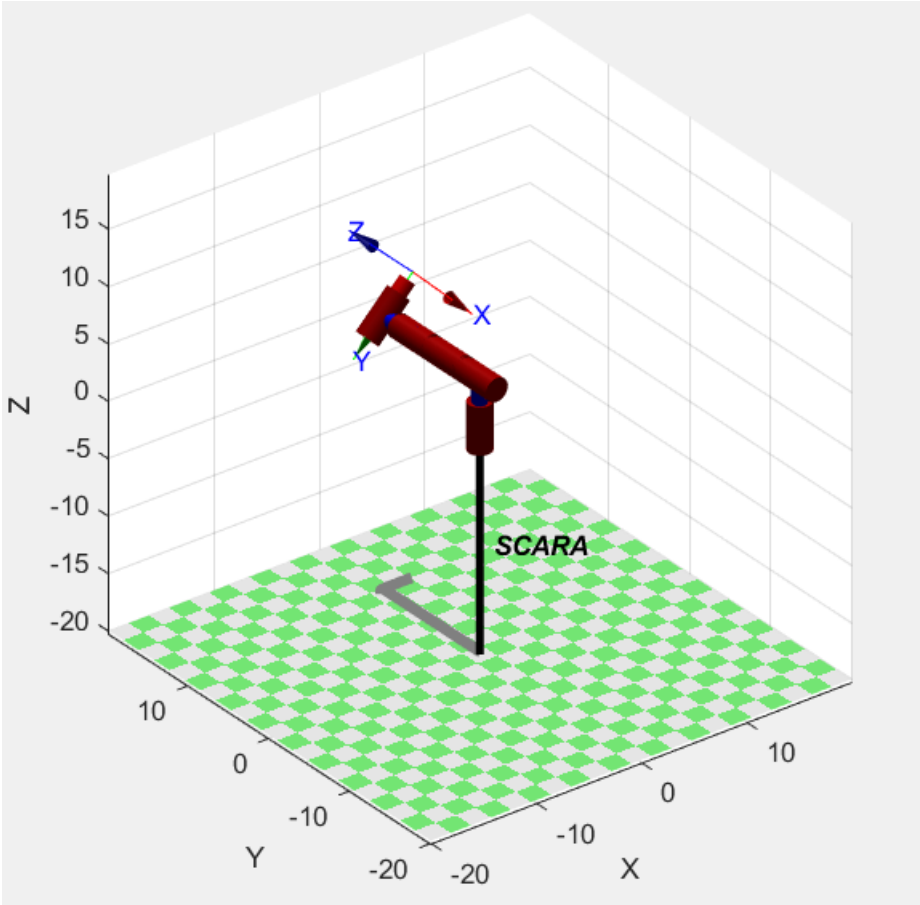
- Teach-pendant: El control del brazo robótico de posiciones y modos se lleva a cabo con un teach-pendant que consta de potenciómetros, switchs y botones que controlan el estado y la posición del brazo.

Modelo

El brazo robótico planteado tendrá 5 grados de libertad que sea capaz de manipular objetos en situaciones que puedan comprometer la seguridad de una persona o supervisión de procesos a distancia.

En nuestro caso hicimos un robot cuyas articulaciones sean de tipo rotacional y el resultado fue el siguiente:


y_1
z_1
x_1
y_2
z_2
x_2
y_3
z_3
x_3
y_4
z_4
x_4
y_0
z_0
x_0


Z
15
10
5
0
-5
-10
-15
-20
Z
X
Y
SCARA
10
0
-10
-20
Y
-20
-10
0
10
X

Cálculo de matriz de Denavit–Hartenberg

Manualmente

Para calcular la cinemática directa de nuestro brazo robótico utilizamos el método de Denavit-Hartemberg, el cual calcula el movimiento de las articulaciones por medio del desplazamiento y rotación en los ejes X y Z y matrices correspondientes de acuerdo con éstos. A continuación, se presentan los desplazamientos de éste.

	Θ	d	a	α
H1	Q1	D1	0	-90
H2	Q2	D2	0	+90
H3	Q3	D3	0	0
H4	Q4	D4	0	0
H5	Q5	D5	0	-90

H1:

$$\begin{bmatrix} \cos(q1) & -\sin(q1) & 0 & 0 \\ \sin(q1) & \cos(q1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-90) & -\sin(-90) & 0 \\ 0 & \sin(-90) & \cos(-90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\cos(q1)} & \mathbf{0} & \mathbf{-\sin(q1)} & \mathbf{0} \\ \boldsymbol{\sin(q1)} & \mathbf{0} & \mathbf{\cos(q1)} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} & \mathbf{0} & \boldsymbol{d_1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

H2:

$$\begin{bmatrix} \cos(q2) & -\sin(q2) & 0 & 0 \\ \sin(q2) & \cos(q2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(90) & -\sin(90) & 0 \\ 0 & \sin(90) & \cos(90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\cos(q2)} & \mathbf{0} & \mathbf{\sin(q2)} & \mathbf{0} \\ \boldsymbol{\sin(q2)} & \mathbf{0} & \mathbf{-\cos(q2)} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \boldsymbol{d_2} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

H3:

$$\begin{bmatrix} \cos(q3) & -\sin(q3) & 0 & 0 \\ \sin(q3) & \cos(q3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\cos(q3)} & \mathbf{0} & \mathbf{\sin(q3)} & \mathbf{0} \\ \boldsymbol{\sin(q3)} & \mathbf{0} & \mathbf{-\cos(q3)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \boldsymbol{d_3} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

H4:

$$\begin{bmatrix} \cos(q4) & -\sin(q4) & 0 & 0 \\ \sin(q4) & \cos(q4) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\cos(q4)} & \mathbf{0} & \mathbf{-\sin(q4)} & \mathbf{0} \\ \boldsymbol{\sin(q4)} & \mathbf{0} & \mathbf{\cos(q4)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \boldsymbol{d_4} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

H5:

$$\begin{bmatrix} \cos(q5) & -\sin(q5) & 0 & 0 \\ \sin(q5) & \cos(q5) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(-90) & -\sin(-90) & 0 \\ 0 & \sin(-90) & \cos(-90) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{\cos(q5)} & \mathbf{0} & \mathbf{-\sin(q5)} & \mathbf{0} \\ \boldsymbol{\sin(q5)} & \mathbf{0} & \mathbf{\cos(q5)} & \mathbf{0} \\ \mathbf{0} & \mathbf{-1} & \mathbf{0} & \boldsymbol{d_5} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

HT:

$$\begin{bmatrix} \cos(q_5)(\cos(q_1)\cos(q_2)\cos(q_3-q_4)-\sin(q_1)\sin(q_3+q_4))+\sin(q_5)(\sin(q_3+q_4)\cos(q_1)\cos(q_2)-\sin(q_1)\cos(q_3-q_4)) & \sin(q_2)(-\cos(q_1)) & \cos(q_5)(\sin(q_3+q_4)\cos(q_1)\cos(q_2)-\sin(q_1)\cos(q_3-q_4))-\sin(q_5)(\cos(q_1)\cos(q_2)\cos(q_3-q_4)-\sin(q_1)\sin(q_3+q_4)) & d_2(-\sin(q_1))+(d_3+d_4)\sin(q_2)\cos(q_1)+d_5\sin(q_2)\cos(q_1) \\ \cos(q_5)(\sin(q_1)\cos(q_2)\cos(q_3-q_4)+\sin(q_3+q_4)\cos(q_1))+\sin(q_5)(\cos(q_1)\cos(q_3-q_4)+\sin(q_1)\sin(q_3+q_4)\cos(q_2)) & -\sin(q_1)\sin(q_2) & \cos(q_3)(\cos(q_1)\cos(q_3-q_4)+\sin(q_1)\sin(q_3+q_4)\cos(q_2))-\sin(q_5)(\sin(q_1)\cos(q_2)\cos(q_3-q_4)+\sin(q_3+q_4)\cos(q_1)) & (d_3+d_4)\sin(q_1)\sin(q_2)+d_5\sin(q_1)\sin(q_2)+d_2\cos(q_1) \\ \sin(q_2)(-\cos(q_3-q_4))\cos(q_5)-\sin(q_3+q_4)\sin(q_5)\sin(q_2) & -\cos(q_2) & \sin(q_2)\sin(q_5)\cos(q_3-q_4)-\sin(q_2)\sin(q_3+q_4)\cos(q_5) & (d_3+d_4)\cos(q_2)+d_5\cos(q_2)+d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matlab

Una vez teniendo el desarrollo de nuestra matriz de cinemática directa del brazo robótico, realizamos el procedimiento en Matlab para comparar nuestros resultados y corroborar los datos obtenidos son correctos y que podemos utilizar nuestro programa para el cálculo de nuestro brazo.

Para este cálculo se realizó una función que calcula la matriz de Denavit-Hartemberg, para cada articulación, a continuación se muestra dicha función:

```
function A = DH(a,alpha,d,theta)
cth = cosd(theta);
sth = sind(theta);
cal = cosd(alpha);
sal = sind(alpha);

Rot_z_theta = [cth -sth 0 0;
               sth  cth 0 0;
                0    0  1 0;
                0    0  0 1];

Trans_z_d = [1 0 0 0;
             0 1 0 0;
             0 0 1 d;
             0 0 0 1];

Trans_x_a = [1 0 0 a;
             0 1 0 0;
             0 0 1 0;
             0 0 0 1];

Rot_x_alpha = [1   0   0   0;
               0  cal -sal 0;
               0  sal  cal 0;
               0   0    0  1];

A = Rot_z_theta*Trans_z_d*Trans_x_a*Rot_x_alpha;
```

Después de hacer la función que calcule la matriz de Denavit-Hartemberg para cada una de las articulaciones.

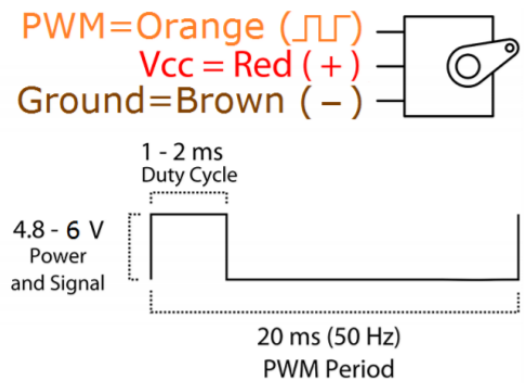
```
%Denavit Hartenberg p/brazo 1
a1 = 0; alpha1 = 0; d1=4; theta1 = 10.8;
H01 = DH(a1,alpha1,d1,theta1); %H^0_1

%Denavit Hartenberg p/brazo 2
a2 = 4; alpha2 = 0; d2=4; theta2 = 9.8;
H12 = DH(a2,alpha2,d2,theta2); %H^1_-2

%Denavit Hartenberg p/brazo -2
a3 = 4; alpha2 = 0; d2=4; theta2 = 10.8;
H13 = DH(a2,alpha2,d2,theta2); %H^1_2

%Denavit Hartenberg p/brazo 4
a4 = 4; alpha2 = 0; d2=4; theta2 = 10.8;
H14 = DH(a2,alpha2,d2,theta2); %H^1_2

%Denavit Hartenberg p/brazo 5
a5 = 4; alpha2 = 0; d2=4; theta2 = 10.8;
H15 = DH(a2,alpha2,d2,theta2); %H^1_2

%Imprimir multiplicación de matrices
display(H01*H12*H13*H14*H15);
```

Y por último la función display nos muestra el resultado en la consola de MatLab de acuerdo con la información dada que se ve en la tabla siguiente:

	Θ	d	a	α
H1	Q1	D1	0	-90
H2	Q2	D2	0	+90
H3	Q3	D3	0	0
H4	Q4	D4	0	0
H5	Q5	D5	0	-90

Cálculo de Potencia

Realizamos para nuestro brazo robótico el respectivo cálculo de potencia del miso, para ello se consideró lo siguiente:



Conexiones para control 1

El torque del motor TowerPro MG90S es de 2.5 kgf*cm y sabemos que la velocidad de operación (velocidad angular) es de 60 grados cada 0.1s. Se calcula en base a 4.8 volts que nos suministra el Arduino.

Para el cálculo de potencia:

$$w = \frac{60\ x \frac{2\pi}{360}}{36} = \frac{10\pi}{3} rad/s$$

La velocidad angular es de $\frac{10\pi}{3} rad/s$ que es igual a 99.99rpm a 4.8 volts de suministro

Obtenemos la potencia en Newton Metro con:

$$M = 1.8\ kg\ cm\ x \frac{1m}{100cm}\ x \frac{9.8m}{s^2} = 0.176\ Nm$$

Calculamos la potencia mecánica

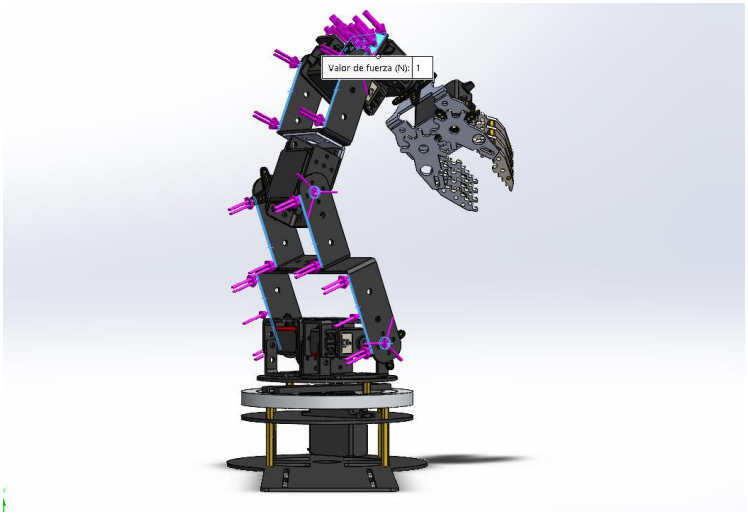
$$W = M\ x\ w$$

$$0.176\ Nm\ x\ \frac{\frac{10\pi}{3} rad}{s} = 1.84 \frac{Nm}{s}$$

La potencia eléctrica nos da = **1.84 Watts**

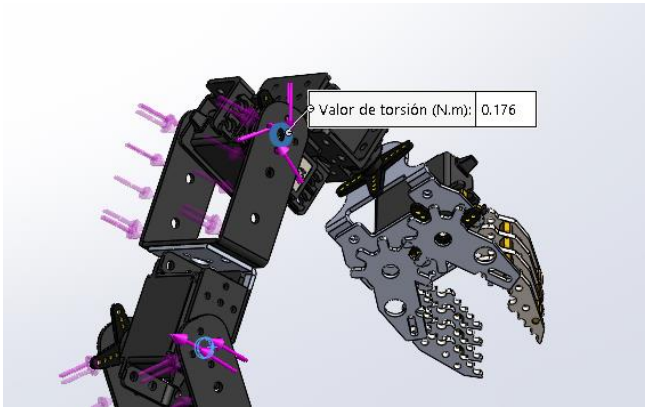
Cálculo de mecánica del Brazo Robótico (Cálculo de Elementos Finitos) (Solid Works)

Se realiza el análisis de elemento finitos a través del parámetro de limite elástico tomando en cuenta las fuerzas de torsión de los servomotores y las fuerzas producida con una carga de 200 gr en griper

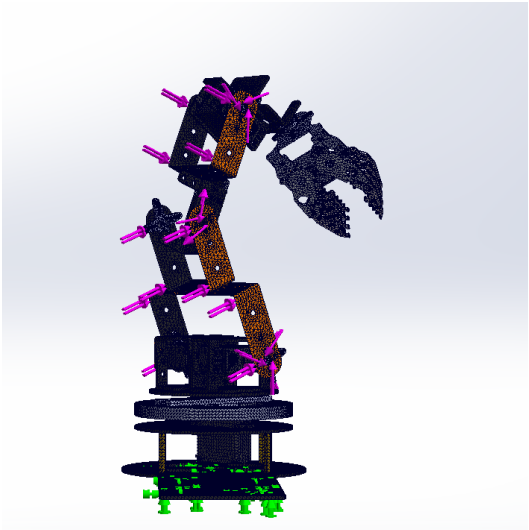


Fuerzas principales en la armadura 1

Observamos que la articulación se ve afectadas por la carga de 200 gr a 1.96 N de fuerza y observamos las fuerzas de torsión conforme al eje de los servomotores

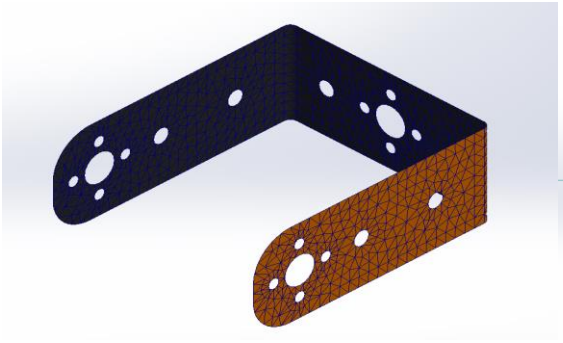


Fuerzas de torsión 1

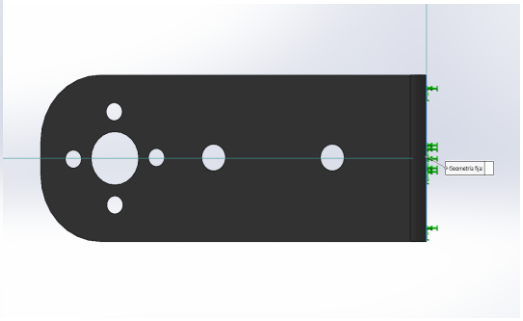


Mallado del sistema 1

La fuerza de torsión (momento) se utilizaron conforme a los cálculos obtenidos del servomotor MG90S de 0.176 Nm, se analizaron cada uno de los elementos conectados a las articulaciones con la siguiente pieza:

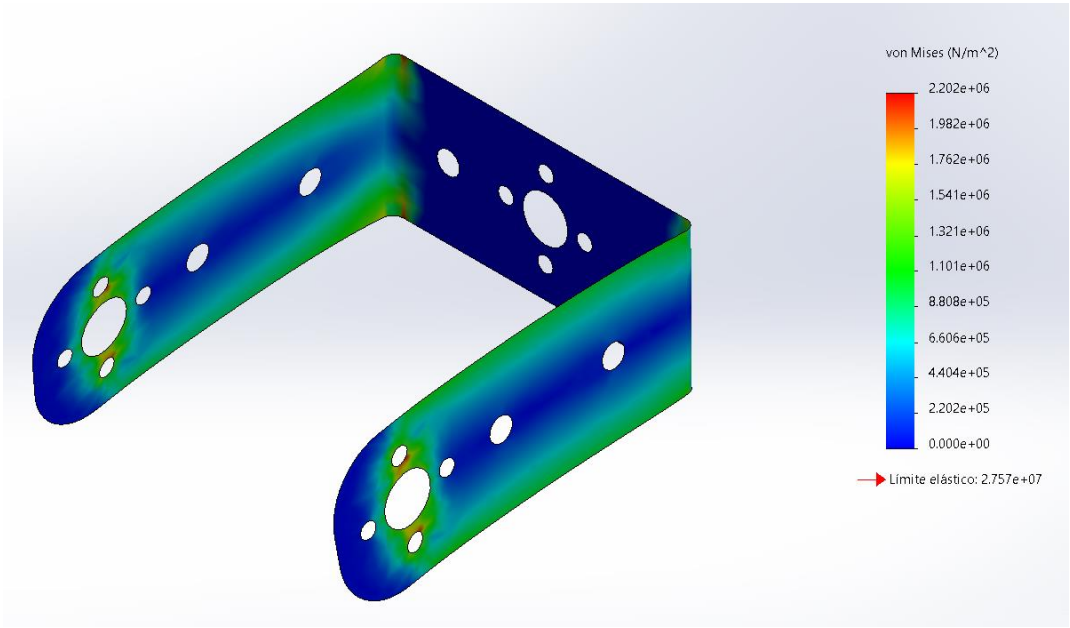


Estalbones del sistema 1



Estalbones del sistema (vista lateral) 2

Al realizar el análisis de torsión y fuerza direccional uniforme, obtuvimos los siguientes resultados



Resultado limite elástico 1

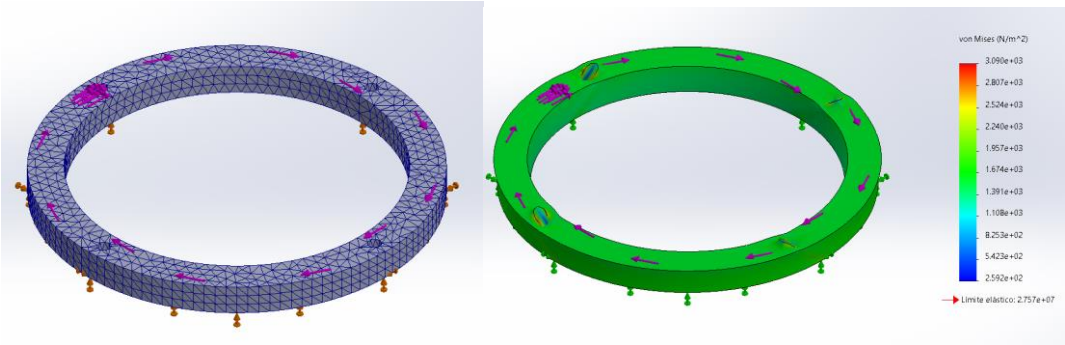
Fuerza de reacción (N)

Componente	Selección	Todo el modelo
Sum X:	0.	-0.0038305
Sum Y:	0.	-0.72591
Sum Z:	0.	-0.0051234
Resultante:	0.	0.72594

Momento de reacción (N.m)

Componente	Selección	Todo el modelo
Sum X:	0.	-0.00093066
Sum Y:	0.	-4.4017E-06
Sum Z:	0.	9.94E-05
Resultante:	0.	0.00093597

Se tomó en cuenta también el anillo de rotación para el primer grado libertad de rotación para la base



Anillo de rotación 1 **Limite elástico**

Conexión de Matlab con Arduino (Comunicación Serial)

Para comunicar nuestro programa de Matlab con el microcontrolador Arduino, es necesario utilizar un protocolo de comunicación serial, el cual por medio de una serie de bits envía datos a través de un canal de comunicación.

Antes de definir el código de comunicación serial tanto en Arduino como en Matlab es necesario establecer un formato a la información que transmitiremos entre ambos, de igual forma se debe establecer el tiempo que durará la transmisión (para este caso 1 segundo) de los datos ya que son parámetros que utilizaremos más adelante en nuestros códigos.

El formato a utilizar para la transmisión de los datos será una cadena con estructura de vector la cual por medio de un carácter separe los 5 ángulos (q1 -q5) que se mandarán desde Matlab hacia Arduino. A continuación, se muestra un ejemplo de un paquete de datos que se enviará.

Si:

q1 = 30.2

q2 = 10.4

q3 = 5.3

q4 = 2.8

q5 = 0.5

La cadena por enviarse será:

30.2/10.4/5.3/2.8/0.5

Una vez definido lo anterior se realizará el código de Matlab para el envío de datos por medio del protocolo de comunicación serial. Para esto lo primero es definir el dispositivo por medio de la función **serialport(string,int)** la cual recibe como parámetro el nombre del puerto por donde saldrá la información y el baudaje (número de bits por segundo) que utilizará la comunicación (el código es para la versión de Matlab r2020b), una vez definido el puerto se utilizará la función **write(device, data, datatype)** donde mandaremos la variable que generó la primera función, luego el conjunto de datos a

mandar y por último el tipo de datos del conjunto escrito. El código para nuestro caso quedó como se muestra a continuación:

```
%Definimos el dispositivo de comunicación serial
device = serialport("COM3",9600);

%Definimos la cadena de información
data = num2str(theta1) + "/" + num2str(theta2) + "/" +
       num2str(theta3) + "/" + num2str(theta4) + "/" + num2str(theta5);

%sacamos la información por el dispositivo previamente definido
write(device, data, "string");
```

Código de comunicación serial Matlab

Después se debe realizar el código en Arduino que reciba la información que sale de la computadora, para esto en el Setup de la Arduino se define el baudaje que tendrá la información recibida.

```
void setup() {
    Serial.begin(9600);
    Serial.setTimeout(1000);
}
```

Código de configuración serial Arduino

Después por medio de la función **Serial.available** establecemos una condición para saber si la Arduino está leyendo datos o no, en caso de que sí estar leyendo datos, se usa la función **Serial.readString()** para recuperar en una variable de cadena la información que entra desde el puerto de comunicación serial y a continuación se separa la información recibida en 5 variables diferentes (q1-q5). Utilizamos una librería llamada StringSplitter.h con el fin de utilizar un método de separación de cadenas.

```
void loop() {
  if (Serial.available() > 0) {
    String data = Serial.readString();
    StringSplitter *splitter = new StringSplitter(data, '/', 5);
    q1 = (splitter->getItemAtIndex(1)).toFloat(); //Angulo Servo 1
    q2 = (splitter->getItemAtIndex(2)).toFloat(); //Angulo Servo 2
    q3 = (splitter->getItemAtIndex(3)).toFloat(); //Angulo Servo 3
    q4 = (splitter->getItemAtIndex(4)).toFloat(); //Angulo Servo 4
    q5 = (splitter->getItemAtIndex(5)).toFloat(); //Angulo Servo 5
    mover();
  }
}
```

Código de lectura y separación de datos Arduino

Desarrollo con Arduino (Tinkercad)

Del funcionamiento

El sistema integrado con Arduino se divide en 3 componentes de gran tamaño. El brazo robótico, sistema IUM y el sistema de control.

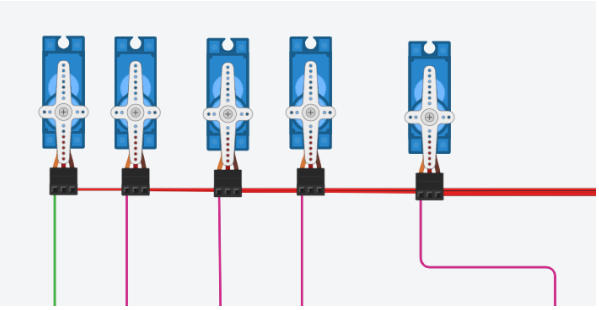


Ilustración 3 Servomotores del sistema.

El brazo robótico consta de 5 servomotores nombrados como base, hombro, codo, muñeca y actuador. Cada uno representa una articulación del brazo robótico, el actuador o efector final tiene un trato especial en los 3 elementos del sistema integrado. Cada servomotor tiene la capacidad de moverse a una posición indicada por un ángulo específico, que puede ser leído a través de los distintos modos del brazo robótico.

En la Ilustración 3 se aprecian de izquierda a derecha los servomotores de base, hombro, codo, muñeca y por último el efector final.

El sistema IUM (Interfaz usuario maquina) esta constituido por una Teachpendant y un display que muestra de manera gráfica el modo y la información en tiempo real del brazo.

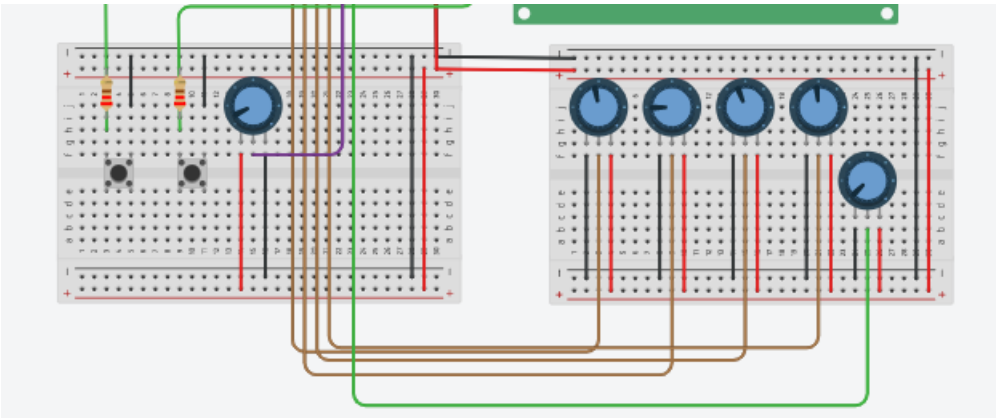


Ilustración 4 Teachpendant

El Teachpendant consta de 6 potenciómetros, dos pushbotton que controlan el modo y las funciones del Teachpendant. En la Ilustración 4 se aprecia el Teachpendant que de izquierda a derecha el primer potenciómetro cambia entre los 3 modos del sistema (automático, manual y serial). Del potenciómetro 2 al 5 controlan en respectivo orden a la base, hombro, codo y muñeca (solo en modo manual). Los pushbotton son para el control fino del efector final, ya que el potenciómetro no otorga un control fino de los motores es por eso que el efector final se controla por medio de pushbutton (solo en modo manual) y cada presión del botón aumenta en un grado la posición del motor. El display tiene dos paginas de muestra de información. Para switchear entre página de información se debe de manipular el switch

que está al lado de del display. La primera muestra el modo en el cual el brazo se encuentra y la segunda pagina de información muestra en tiempo real la posición de cada servomotor.

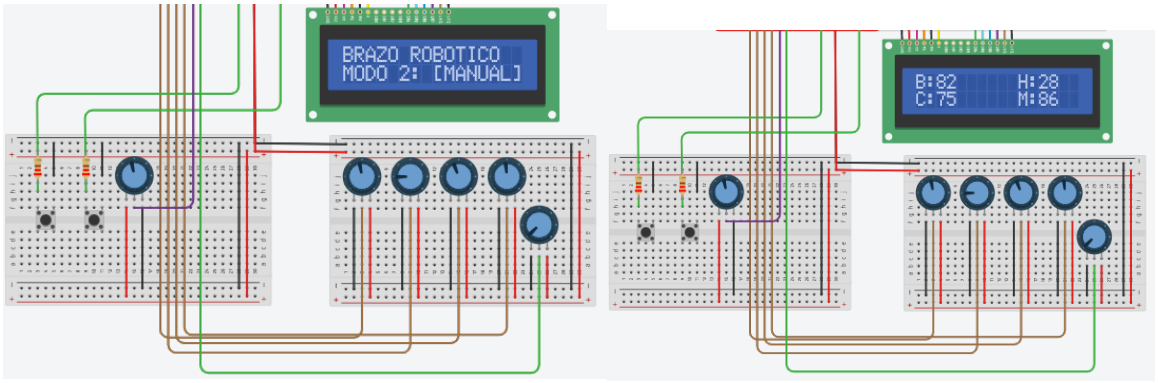


Ilustración 5 Display en mostrando el modo y valores de ángulo.

El sistema de control hace referencia a la codificación que le da vida al sistema general en si. El código esta desarrollado en C en el IDE Arduino y cargado en un microcontrolador Arduino Uno.

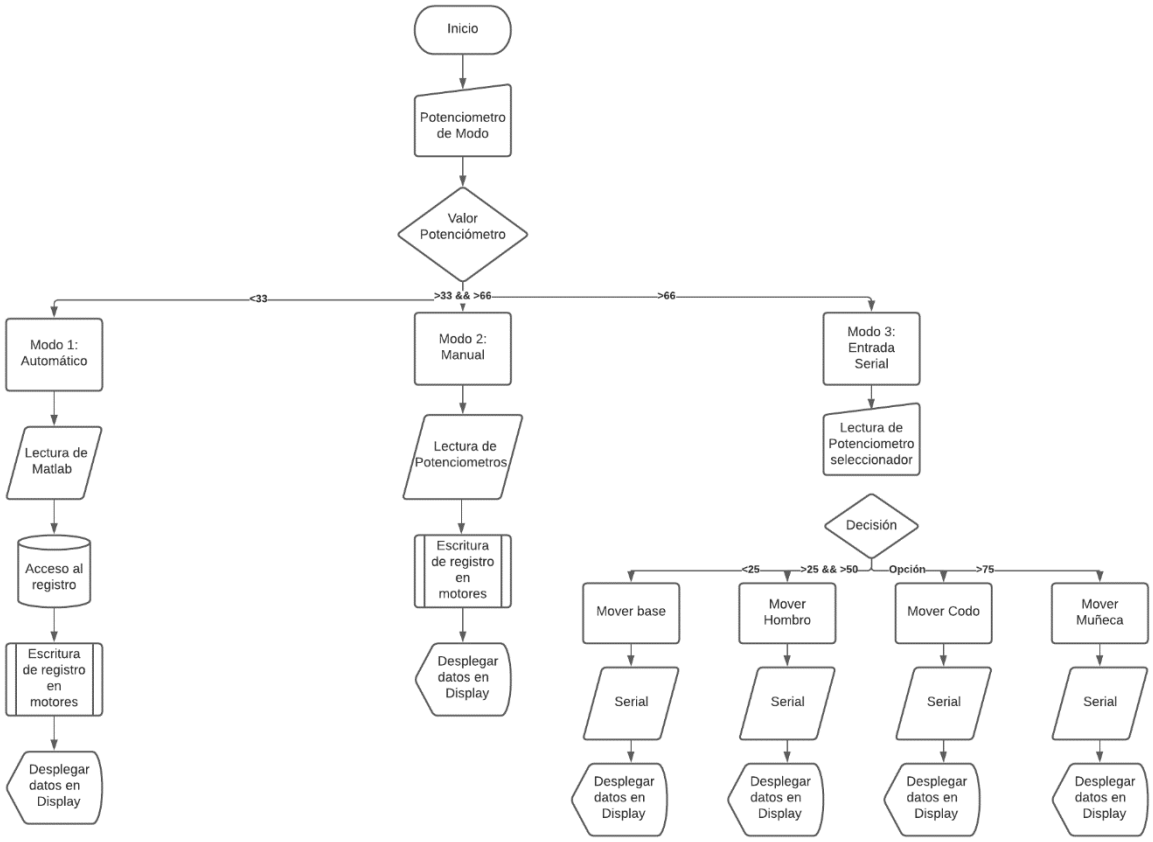


Ilustración 6 Diagrama de Bloques de codificación

De la Interfaz

La interfaz del sistema es diseñada para favorecer la interacción sencilla entre el brazo robótico y el usuario. Hay tres subinterfaces que se pueden definir como modos de interacción. La manera de cambiar entre modo de accionamiento es mediante un potenciómetro instalado en nuestro teachpendant. El primer modo es un modo automático en donde los valores de posición de cada motor son adquiridos de manera automática a través de la conexión serial con Matlab. Matlab envía a Arduino los valores de la posición final del actuador y el sistema hace la combinación necesaria entre los motores para llegar a esa posición. El usuario en este modo no tiene manera de controlar por medio del Teach-pendant el brazo robótico, lo cual es una ventaja ya que no hay manera en que el operador interfiera con el sistema de control automático. Mediante el LCD se muestran los ángulos en tiempo real en los cuales el brazo se encuentra, lo cual es demasiado útil ya que de manera instantánea los valores de posición se refrescan de acuerdo a los valores del motor. En este modo únicamente se modificarán los valores del display cuando en Matlab se modifique la posición deseada.

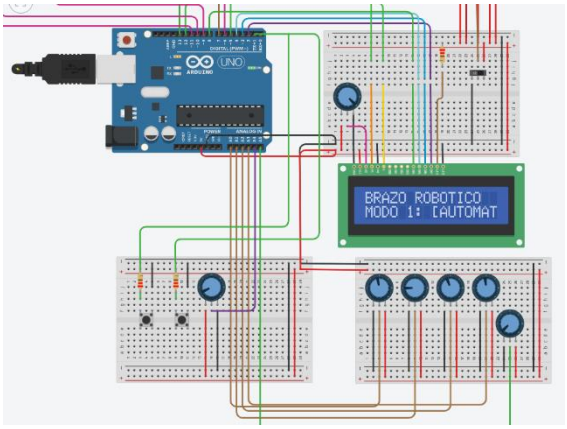


Ilustración 7 Modo Automático

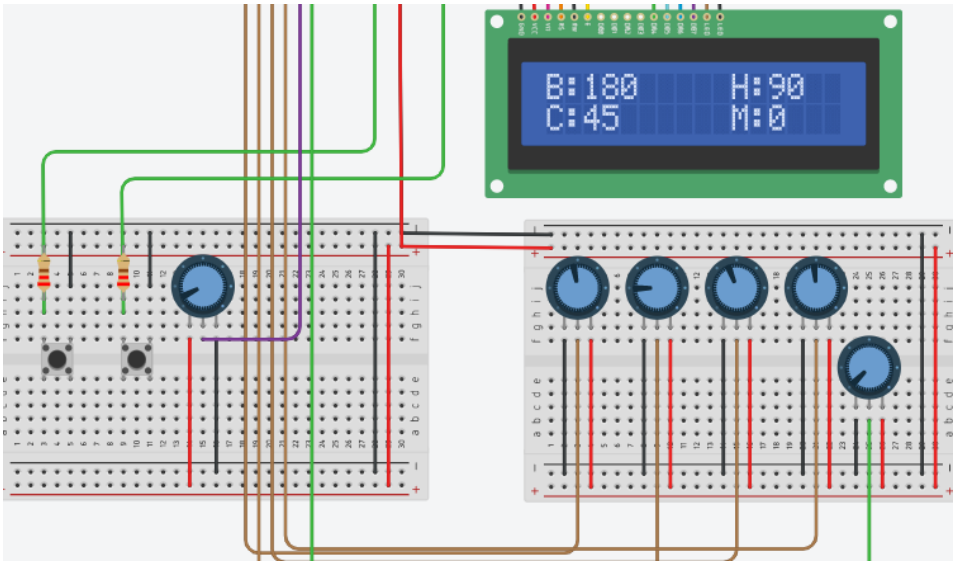


Ilustración 8 Ángulos del modo automático

El segundo modo de operacoón es el modo manual que habilita al 100% la funcionalidad del Teachpendant. El potenciometro se recorre a la mitad de su recorrido y de esta manera se entra al segundo modo de interacción del brazo robótico. En este modo los motores y el teach pendant están totalemnete liberados para poder ser manipulados a gusto por el susario. Los potenciometros moveran los motores de base, hombro, codo y muñeca. Mientras que los pushbotton moveran mediante pequeños movimientos muy finos al actuador final.

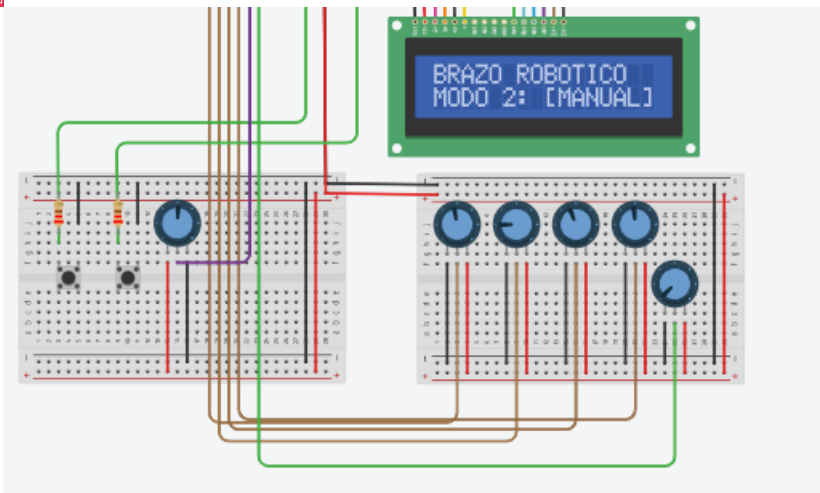


Ilustración 9 Modo Manual

En la Ilustración 8 se muestra el cambio a modo manual desplegado por el Display que nos muestra que el brazo robótico está listo para ser operado mediante el teachpendant y que la lectura por Matlab y puerto serial estan totalmente deshabilitadas.

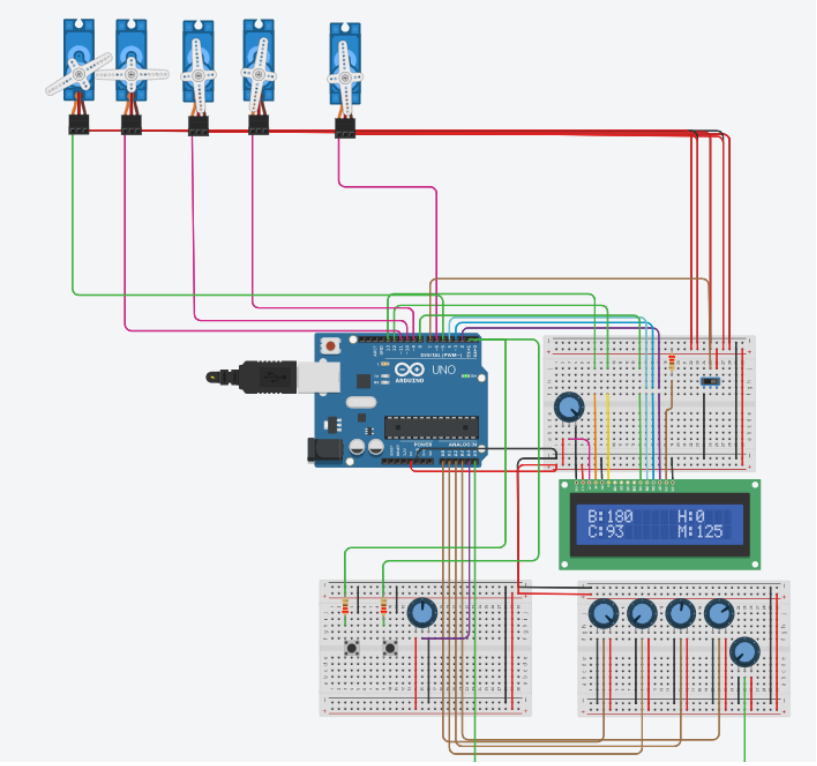


Ilustración 10 Movimiento de motores en modo manual

En la Ilustración 9 se puede apreciar los ángulos desplegados por el display. Los ángulos mostrados representan la posición exacta de cada motor del brazo robótico. Así como el movimiento de los potenciómetros muestra un cambio instantáneo tanto en la posición del motor como el en el ángulo mostrado por el display. Cabe mencionar que solamente se muestran los valores de posición de la base,

hombro, codo y muñeca mientras que el del efector final no es desplegado en ningún modo debido a que su movimiento fino deberá ser controlado de manera visual.

El tercer y último modo de funcionamiento deshabilita el control manual y el control por entrada serial de Matlab. Lo que quiere decir que si se mueven los potenciómetros de control de posición nada pasara en el sistema ya que estos están bloqueados. Este modo habilita la escritura de ángulos por entrada de teclado, para seleccionar el motor que se desea mover es necesario mover el último potenciómetro del teachpendant. Solo estarán disponibles los motores de base, hombro, codo y muñeca. Para saber que motor está seleccionado se deberá abrir la consola serial de Arduino en donde se mostrará el mensaje de “Moviendo base…..” en caso de tener la base seleccionada con el potenciómetro de selección. Una vez teniendo seleccionado el motor que se moverá y habiendo comprobado que el monitor serial muestra que se tiene seleccionado el motor correcto solo se deberá escribir en la consola el ángulo al cuál queremos mover el motor, de inmediato aparecerá el ángulo de movimiento en el monitor serial y el motor comenzará a moverse a la posición solicitada, el display mostrará este cambio de inmediato.

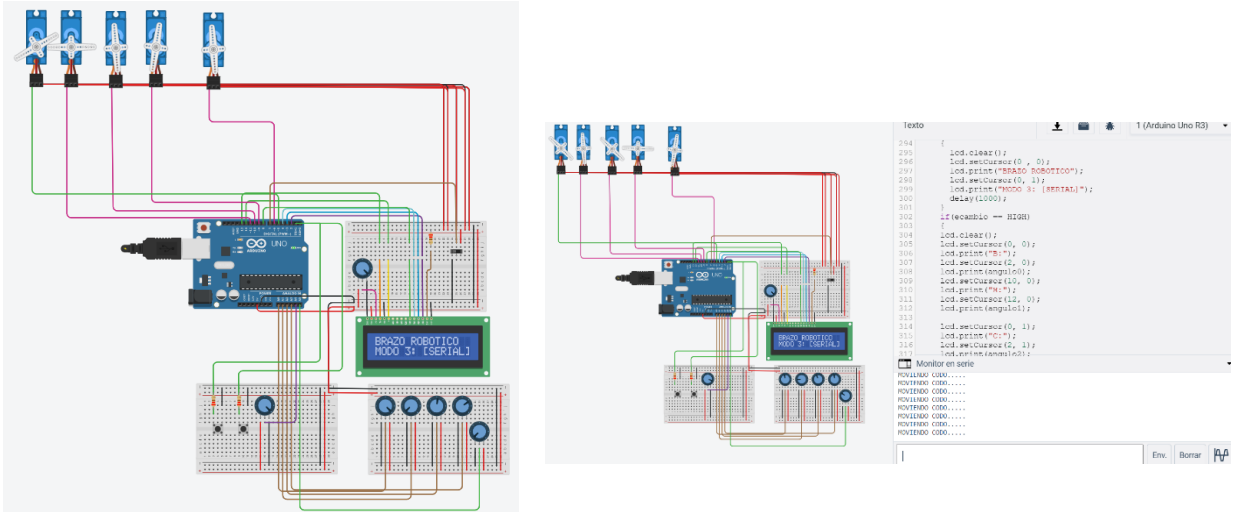


Ilustración 11 Modo serial

Del esquemático de conexiones

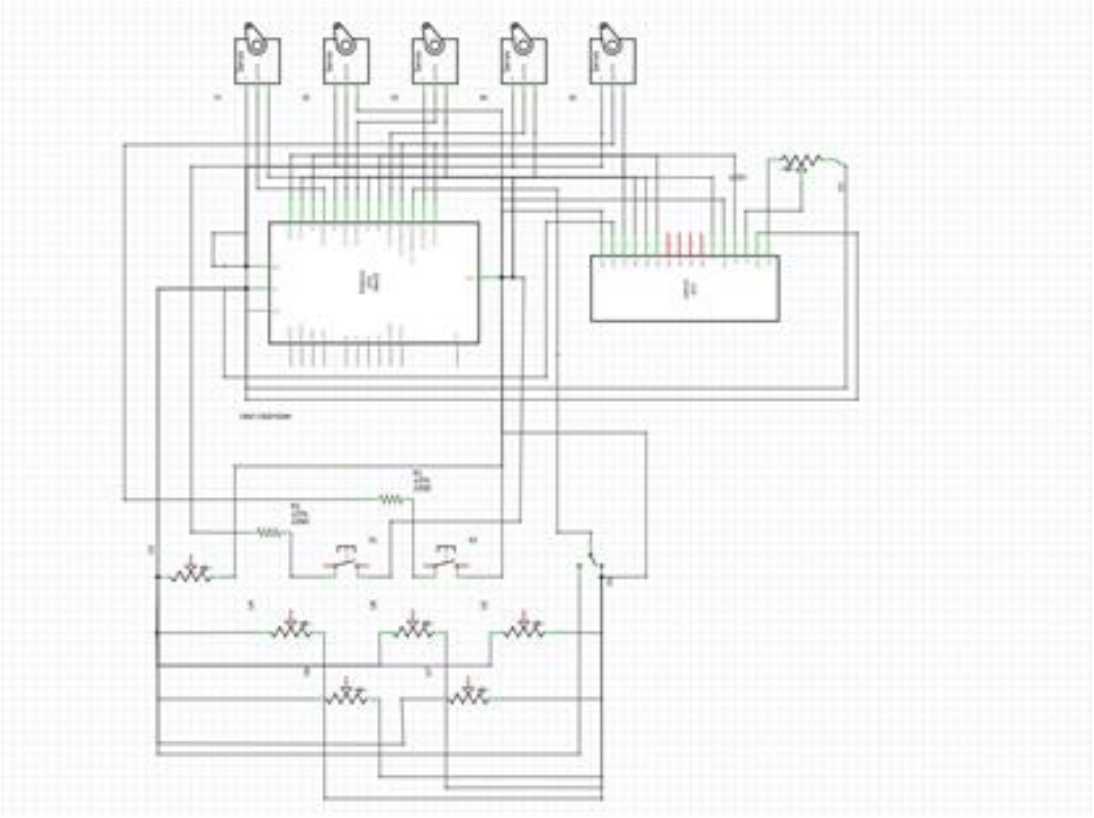


Ilustración 12 Esquemático de conexiones

En la Ilustración 11 se muestra el plano de conexiones eléctricas que integran todo el sistema. El microprocesador Arduino Uno funge como parte central del sistema que analiza y procesa las señales de entrada y reacciona en señales de salida para el sistema. Los elementos de salida son servomotores cuya conexión se hace a puertos digitales/pwm del Arduino, el resto de sus conexiones es a voltaje y a puesta a tierra. Los servomotores son el único elemento de salida con los que el sistema cuenta ya que la posición del brazo robótico es la variable de control deseada.

El elemento de entrada es el teachpendant junto a todos sus botones y potenciómetros. Estos darán las señales de entrada que el microprocesador analizará y sus señales de salida irán en función a estos. Los potenciómetros son conectados a puertos analógicos mientras que los interruptores son conectados a puertos digitales.

El display siendo un elemento de la IUM se conecta a pines digitales para su control.

Costo

Una vez contemplado todo lo anterior procedimos a el cálculo del costo del robot, para ello realizamos la siguiente tabla:

Materiales	Cantidad	Costo	Total
Suporte tipo L	1	80	80
Suporte tipo C	3	100	300
Suporte universal	4	120	480
Servomotor TowerPro	6	125	750
Disco de sujeción	6	40	240
Gripper de Metal para Servo MG995	1	320	320
Total			2170

Referencias

C. Chen, R. Hong and H. Wang, "Design of a Controlled Robotic Arm", 2016 3rd International Conference on Green Technology and Sustainable Development (GTSD), *Kaohsiung, 2016.*

C. Chen, Z. Chen, "A Remote Controlled Robotic Arm That Reads Barcodes and Handles Products", Deparment of Mechanical Engenireen, National Kaohsiung University of Science and Technology, Kaohsiung, 2018.

Vasco F, "Brazo Robótico de 5 grados de libertad con transmission de video e interfaz de control inalámbrico", Universidad de San Buenaventura, Medellín, 2014.

Ollero, A. "Robótica: Manipuladores y Robots Móviles", Marcómbo S.A. Barcelona, 2001.

Electronicoscalda. (_). Datasheet MG90S. 2020, de TOWER PRO Sitio web: https://www.electronicoscaldas.com/datasheet/MG90S_Tower-Pro.pdf

Luis del Valle Hernández. (2016). Servomotor con Arduino tutorial de programación paso a paso. 07/12/2020, de Progamación facil Sitio web: https://programarfacil.com/tutoriales/fragmentos/servomotor-con-arduino