

# A Focus on Motivation and Learning Enhancement in Beginning Engineering Students

## *A comparative experimentation in robotic arm control using Matlab and Arduino.*

Juan F. Tisza C., DSc.<sup>1,2</sup>(c) , Alessandra Orejon BSc.<sup>2</sup>, Carl Castañeda BSc.<sup>2</sup>, Luis Arteaga BSc.<sup>2</sup>, Camila Zuñe BSc.<sup>2</sup>, Angelo Garcia BSc.<sup>2</sup>

<sup>1</sup>Universidad Nacional de Ingeniería, Perú, <sup>2</sup>Universidad Nacional Mayor de San Marcos, Perú,  
[jtisza@uni.edu.pe](mailto:jtisza@uni.edu.pe), [alessandra.orejon@unmsm.edu.pe](mailto:alessandra.orejon@unmsm.edu.pe), [carl.castaneda@unmsm.edu.pe](mailto:carl.castaneda@unmsm.edu.pe), [luis.arteaga9@unmsms.edu.pe](mailto:luis.arteaga9@unmsms.edu.pe),  
[camila.zune@unmsm.edu.pe](mailto:camila.zune@unmsm.edu.pe), [angelo.garcia@unmsm.edu.pe](mailto:angelo.garcia@unmsm.edu.pe)

**Abstract** – *This article presents a proposal for improvement in the teaching-learning process and in the motivation of engineering students, through two forms of simulation of the control of a robotic arm of articulated structure with rotary movement and linear displacement. Matlab, Arduino and the graphical interface of the CoppeliaSim program are used. In Matlab you use Simulink with the Simscape Multibody Link Plugin tool and with the GUI function the control of the robotic arm is developed with different values in the angles of the joints, to finally simulate the movements. On the other hand with the Arduino 1 complemented with the CoppeliaSim software the movements of the arm are visualized. In this way, the comparison of the two ways of implementing the solutions is made. The results are presented in a comparative table that evaluates the control and simulation to determine the most advantageous solution. The validation in the application of the proposal is presented in bar charts that indicate the degree of satisfaction obtained in a sample of 35 students.*

**Keywords**-- *Control, Experiential learning, Matlab, Arduino, robotic arm.*

**Digital Object Identifier:** (only for full papers, inserted by LACCEI).

**ISSN, ISBN:** (to be inserted by LACCEI).

**DO NOT REMOVE**

# Un Enfoque de Motivación y Mejora del Aprendizaje en Estudiantes Principiantes de Ingeniería

## *Una experimentación comparativa en control de brazo robótico usando Matlab y Arduino.*

Juan F. Tisza C., DSc.<sup>1,2(c)</sup> , Alessandra Orejon BSc.<sup>2</sup>, Carl Castañeda BSc.<sup>2</sup>, Luis Arteaga BSc.<sup>2</sup>, Camila Zuñe BSc.<sup>2</sup>, Angelo Garcia BSc.<sup>2</sup>

<sup>1</sup>Universidad Nacional de Ingeniería, Perú, <sup>2</sup>Universidad Nacional Mayor de San Marcos, Perú,  
[jtisza@uni.edu.pe](mailto:jtisza@uni.edu.pe), [alessandra.orejon@unmsm.edu.pe](mailto:alessandra.orejon@unmsm.edu.pe), [carl.castaneda@unmsm.edu.pe](mailto:carl.castaneda@unmsm.edu.pe), [luis.arteaga9@unmsms.edu.pe](mailto:luis.arteaga9@unmsms.edu.pe),  
[camila.zune@unmsm.edu.pe](mailto:camila.zune@unmsm.edu.pe), [angelo.garcia@unmsm.edu.pe](mailto:angelo.garcia@unmsm.edu.pe)

**Resumen–** En este artículo se presenta una propuesta de mejora en el proceso de enseñanza-aprendizaje y en la motivación de los estudiantes de ingeniería, mediante dos formas de simulación del control de un brazo robótico de estructura articulada con movimiento rotativo y desplazamiento lineal. Se utilizan los softwares Matlab, Arduino y la interfaz gráfica del programa CoppeliaSim. En Matlab se usa el Simulink con la herramienta Simscape Multibody Link Plugin y con la función GUI se desarrolla el control del brazo robótico con distintos valores en los ángulos de las articulaciones, para finalmente simular los movimientos. Por otro lado con el Arduino 1 complementado con el software CoppeliaSim se visualiza los movimientos del brazo. De esta manera se realiza la comparación de las dos formas de implementar las soluciones. Los resultados son presentados en una tabla comparativa que evalúa el control y la simulación para determinar la solución más ventajosa. La validación en la aplicación de la propuesta, es presentada en diagramas de barras que indican el grado de satisfacción obtenido en una muestra de 35 estudiantes.

**Palabras claves--** Control, Aprendizaje experimental, Matlab, Arduino, brazo robótico.

### I. INTRODUCCIÓN

Actualmente existe la necesidad de mejorar el nivel de aprendizaje y la motivación de los estudiantes de ingeniería de los primeros ciclos de estudio a fin de garantizar el éxito en la formación profesional y reducir la deserción durante el proceso formativo. En este contexto se implementa el trabajo de investigación que se presenta en este artículo.

El objetivo principal es que el estudiante de ingeniería de

los primeros niveles empiece a experimentar a través de la manipulación y el control de un brazo robótico a fin de que esta experiencia lo motive en la adquisición de más conocimientos y habilidades para el estudio e investigación en ingeniería y en robótica, dichos conocimientos serán profundizados, en el desarrollo de las siguientes asignaturas que contemplan los planes de estudios correspondientes; para conseguir los resultados que se buscan, se implementan dos formas de control y manipulación del brazo robótico. La idea central es que el estudiante pueda operar el control sin ingresar a un tratamiento profundo de la robótica, por lo que las explicaciones a los estudiantes en este nivel se basan mayormente en diagrama de bloques y en forma cualitativa.

En el trabajo se busca observar y evaluar en forma comparada los principales beneficios de usar Matlab o Arduino para mejorar el proceso enseñanza-aprendizaje.

Matlab y Arduino son herramientas informáticas muy usadas en el ámbito académico a nivel internacional y son programas básicos para realizar una diversidad de proyectos.

En base a las justificaciones antes indicadas, se resume tres objetivos específicos que establece la propuesta presentada:

- Desarrollar e implementar dos formas experimentales de aprendizaje orientado a estudiantes de los primeros niveles de ingeniería, cuya validación aplicativa se realiza en una muestra de 35 estudiantes de ingeniería electrónica.
- Comparar las soluciones (con Matlab y Arduino) aplicado al sistema de control del brazo robótico, y las simulaciones.
- Presentar tabla comparativa de beneficios que se obtienen entre las dos formas de implementación

La robótica es una tecnología innovadora y emergente con diversas posibilidades de utilización en las actividades

**Digital Object Identifier:** (only for full papers, inserted by LACCEI).  
**ISSN, ISBN:** (to be inserted by LACCEI).  
**DO NOT REMOVE**

productivas y también en el ámbito académico como herramienta de aprendizaje. El interés por la robótica educativa en los últimos años ha ido en aumento y constituye una integración de conocimientos (teóricos y prácticos) incluyendo la programación [03].

Los robots articulados se encuentran entre los robots más comunes utilizados en la actualidad. Buscan asemejarse al brazo humano y por eso se le denomina brazo robótico o brazo manipulador [04]. Se describe el brazo robótico como una cadena de eslabones que se mueven mediante articulaciones que son accionadas por motores.

La aplicación de la robótica combinada con programas de ayuda permiten mejorar el diseño de sistemas y eliminar errores antes del desarrollo de prototipos físicos tanto a nivel académico como profesional. Como antecedentes de esta investigación tenemos la publicación de Lopez y colaboradores. [5] quien propuso un sistema de control que permite sincronizar los estados del sistema caótico MACM con las velocidades lineales y angulares de un robot móvil diferencial para inducir dinámicas caóticas con el programa Matlab. En [7] se introdujo a estudiantes de Tecnología en Electromecánica de la Escuela de Formación de Tecnólogos (ESFOT) al campo de la robótica, mediante la simulación de un robot con 6 grados de libertad, utilizando el Toolbox Robotics de Matlab.

Por otro lado tenemos el uso de herramientas tales como Arduino que permiten el trabajo dinámico de componentes de alta complejidad en la construcción de sistemas interactivos. La temática que se trata en este artículo es de interés en el ámbito académico y educativo, en este sentido tenemos la publicación de Burbano, Cárdenas y colaboradores. [06] quienes propusieron el desarrollo de un robot móvil con un sistema de recolección de objetos controlados remotamente para fines didácticos. Otro trabajo a mencionar es el realizado por Vicente y colaboradores.[8] quien propuso comprobar la capacidad de producir un robot de bajo coste sin perjudicar la precisión que pueda alcanzar, en dicha investigación se seleccionó el robot SCARA 3RRR.

## II. METODOLOGÍA

A continuación se describen las fases metodológicas que se siguen para desarrollar las dos soluciones propuestas para el control de un brazo robótico de estructura articulada con movimiento rotativo y desplazamiento lineal y se ensaya mediante la experimentación virtual a través de las simulaciones de sus movimientos.

En primer lugar se investigó los tipos de proyectos que se desarrollan utilizando la estructura de un brazo robótico en el cual se identificó las características generales y estructurales del brazo robótico, a continuación se recopiló la información de una variedad de robots, tanto en su funcionamiento como en su ensamblaje.

En la segunda fase se investigó la aplicación orientada a la mejora del aprendizaje, para aplicar posteriormente un

análisis comparativo en la implementación de los controles con Matlab y Arduino.

En la tercera fase se decidió integrar el programa Coppelia Sim para tener una interfaz del brazo robótico el cual brinda una mejor óptica para el despliegue de movimientos del robot, dicho programa permite desarrollar la interfaz gráfica para la codificación de Arduino, debido a que en Arduino no es posible visualizar la simulación de robots en la pantalla digital.

En la cuarta fase se recopilamos los datos programables para el Arduino, incorporando sus librerías para aplicarlos en los requerimientos de temporización interna en la activación de servomotores, se utilizó las librerías servo.h que permitieron activar hasta doce servomotores.

En la quinta fase se seleccionó el modelo de brazo robótico de Solidworks, para ser importado hacia Matlab y poder aplicar el control a través de los códigos de programación, posteriormente se desarrolló la programación de la interfaz gráfica de usuario, incluyendo los cálculos numéricos para controlar el movimiento del brazo robótico.

En la sexta y última fase se logró recopilar y estructurar la programación del Matlab y Arduino, es decir se pudo hacer funcionar las simulaciones de las gráficas del brazo robótico en ambos software en Matlab empleando la herramienta de solidwork para obtener las piezas del brazo robótico y en Arduino empleando de soporte la interfaz gráfica de CoppeliaSim.

## III. DESARROLLO

### A. Software, materiales y herramientas utilizadas

- Software Matlab 2020b
- Arduino UNO
- Programa CoppeliaSim

### B. Esquemas - Topología

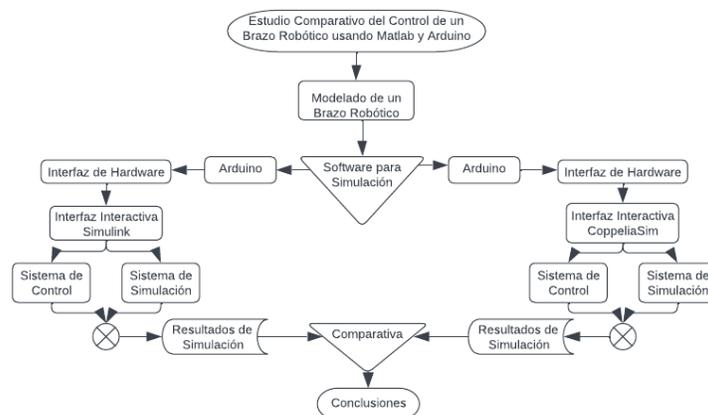


Fig. 1 Diagrama de flujo de estructura del sistema  
Fuente. Elaboración propia

### C. Procedimiento

1) Configuración de códigos preestablecidos para Matlab basado en los principios y bases teóricas:

- Matlab

La versión utilizada para las simulaciones es la 2020b, mediante el uso del plug-in Simscape Multibody Link, que permite exportar ensambles desde SolidWorks, generando un archivo XML que puede ser importado a Simscape Multibody, el cual es un entorno de simulación para sistemas mecánicos 3D.

- Configuración de Simulink

Simulink es un entorno que permite el diseño de sistemas mediante el uso de bloques, siendo Simscape un entorno de modelamiento físico, parte de Simulink. Se importa el modelo mediante el uso de la función `smimport` y se genera automáticamente el modelo mecánico del brazo robótico. Este modelo no posee control alguno, siendo necesario programar y añadir los bloques de control correspondientes.

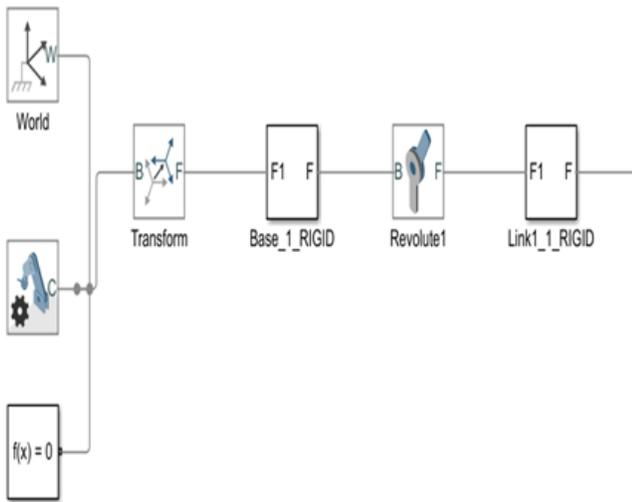


Fig. 2 Modelo mecánico del brazo robótico generado por la función `smimport`.

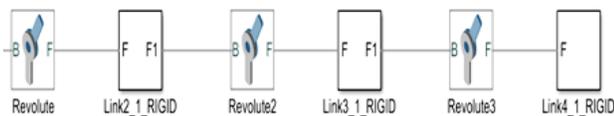


Fig. 3 Continuación del modelo mecánico del brazo robótico generado por la función `smimport`.

Se modifica el parámetro Motion de tal forma que permita el ingreso a través de una entrada con el fin de controlar los ángulos en los bloques Revolute, Revolute1, Revolute2 y Revolute3.

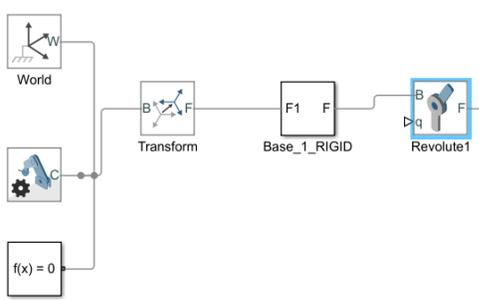


Fig. 4 Ajuste de parámetros del bloque Revolute1

Para ingresar los ángulos se usan tres bloques:

- Constant: Tiene como salida una constante especificada por el usuario.
- Slider Gain: Es un bloque que aplica una ganancia variable al valor de la constante.
- Simulink-PS Converter: Convierte una señal de Simulink a una señal física que puede ser interpretada por los bloques Revolute.

Se ingresa el valor 1 para el bloque Constant, el rango de ganancia de -180 a 180 para el bloque Slider Gain y la unidad de grados sexagesimales para el bloque Simulink-PS Converter. Con esta secuencia se puede ingresar un rango de valores de  $-180^\circ$  a  $180^\circ$  en los bloques Revolute.

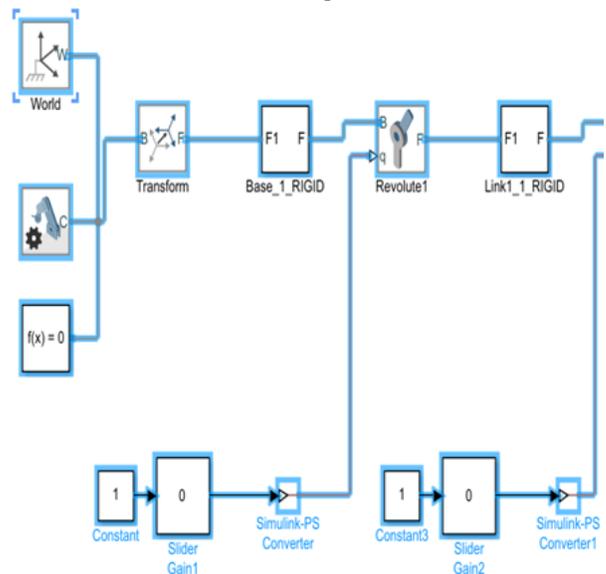


Fig. 5 Modelo mecánico del brazo robótico generado con entradas de ángulos.

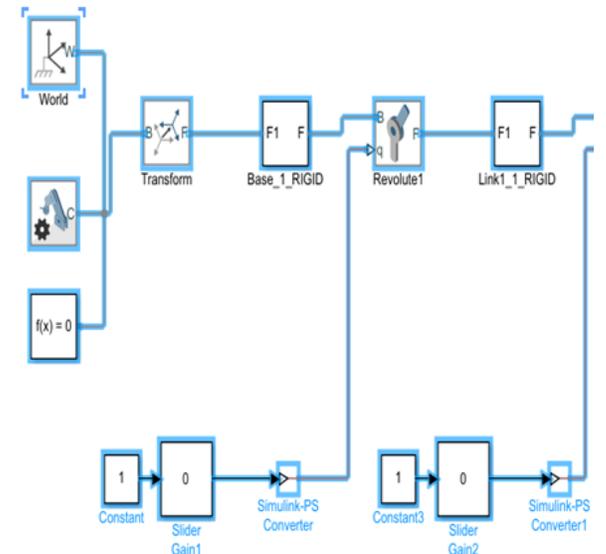


Fig. 6 Continuación del modelo mecánico del brazo robótico generado con entradas de ángulos.

- Control del brazo robótico mediante Matlab GUI

El control de un brazo robótico se puede realizar mediante el uso de dos técnicas conocidas como cinemática directa y cinemática inversa. La cinemática directa permite determinar la posición del extremo final del brazo robótico mediante el ingreso de los ángulos de cada articulación. Por otro lado, la cinemática inversa permite determinar los ángulos requeridos de cada articulación para que el extremo final del brazo esté en una posición determinada por el usuario. En este trabajo sólo se tomará un enfoque de control mediante el uso de cinemática directa.

Para controlar los movimientos del brazo robótico se utiliza una interfaz gráfica de usuario (GUI) programada en Matlab que permite modificar los valores de ganancia del bloque Slider Gain y calcular la posición del extremo final del brazo robótico.

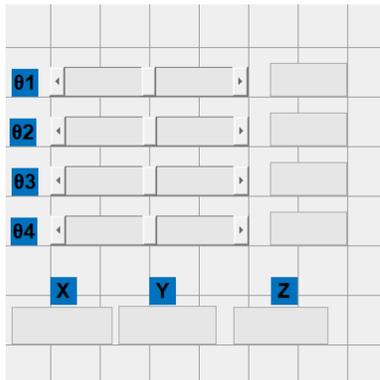


Fig. 7 Ventana de diseño-GUI del brazo robótico.

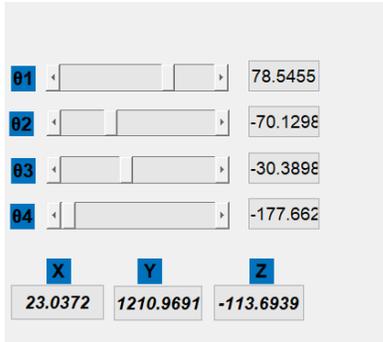


Fig. 8 Ventana de ajuste-GUI del brazo robótico

La GUI contiene tres componentes que se indican en la tabla I:

TABLA I  
COMPONENTES Y DESCRIPCIÓN DE LA GUI

Componente	Descripción
slider	Permite representar un rango de valores
static text	Caja que muestra un string de texto
editable text	Caja de texto editable

El componente slider es usado para representar los valores de las ganancias de los bloques Slider Gain en un rango de -180 a +180, el componente editable text es usado para mostrar al usuario los valores de los slider y la posición del extremo final. Por último el componente static text se usó para etiquetar a que variable representa cada componente y los valores mostrados por los componentes slider y editable text.

- Programación de las funciones asociadas al GUI

Mediante el uso de los comandos callback se asociaron funciones a cada componente de la GUI.

La función asignada a los slider se encarga de registrar los valores de cada slider, interpretarlos como ángulos de las articulaciones y registrando estos valores en los editable text asociados a cada articulación. A continuación un extracto de código donde se muestran las funciones encargadas de la adquisición de ángulos y el registro de estos en la GUI.

%Adquisición de ángulos y registro en la GUI

```
thetal=get(handles.slider1,'value');
set(handles.edit2,'string',num2str(thetal));
theta2=get(handles.slider2,'value');
set(handles.edit3,'string',num2str(theta2));
theta3=get(handles.slider3,'value');
set(handles.edit4,'string',num2str(theta3));
theta4=get(handles.slider4,'value');
set(handles.edit5,'string',num2str(theta4));
```

Después de realizar lo anterior, la función le asigna valores de los ángulos en los bloques Slider Gain.

\$Ingreso de valores en bloques Slider Gain

```
ser_param([modelName'/sliderGain'],'Gain',num2str(thetal));
ser_param([modelName'/sliderGain2'],'Gain',num2str(theta2));
ser_param([modelName'/sliderGain3'],'Gain',num2str(theta3));
ser_param([modelName'/sliderGain4'],'Gain',num2str(theta4));
```

Para concluir, se calcula la posición del brazo robótico y se registran las coordenadas en la GUI. Después de los procedimientos indicados en las figuras 9 y 10, se procedió a ejecutar las simulaciones con distintos valores en los ángulos de las articulaciones.

2) Configuración de códigos preestablecidos para arduino, basados en los principios y bases teóricas:

- En el Arduino Uno

Se utiliza la placa Arduino Uno ya que es compatible para activar los servomotores conectado a una fuente de alimentación DC de 7v a 12v con al menos una corriente de 750 mA. Debido a que el Arduino Uno es una placa basada en el microcontrolador ATmega328 que tiene:

- 6 entradas analógicas
- 14 salidas digitales
- Conector de alimentación
- Conector a puerto USB

A continuación un extracto de código donde se muestran las funciones encargadas de la adquisición de ángulos y el registro de estos en la GUI.

```
%Cinematica directa
T1 = [ cosd(theta1) 0 -sind(theta1)0;
      sind(theta1) 0 -cosd(theta1)0;
      0 -1 0 486;
      0 0 0 1];
T2 = [ cosd(theta2)-sind(theta2)0;a2*cosd(theta2);
      sind(theta2)-cosd(theta2)0;a2*sind(theta2);
      0 0 1 0;
      0 0 0 1];
T3 = [ cosd(theta3)-sind(theta3)0;a3*cosd(theta3);
      sind(theta3)-cosd(theta3)0;a3*sind(theta3);
      0 0 1 0;
      0 0 0 1];
T4 = [ cosd(theta4)-sind(theta4)0;a4*cosd(theta4);
      sind(theta4)-cosd(theta4)0;a4*sind(theta4);
      0 0 1 0;
      0 0 0 1];
T= T1*T2*T3*T4;
Px=(1,4);
PY=T(2,4);
Pz=T(3,4);
%Registro de coordenadas xyz en la GUI
set(handles.edit6,'strino',num2str(Px));
set(handles.edit7,'strino',num2str(Py));
set(handles.edit8,'strino',num2str(Pz));
```

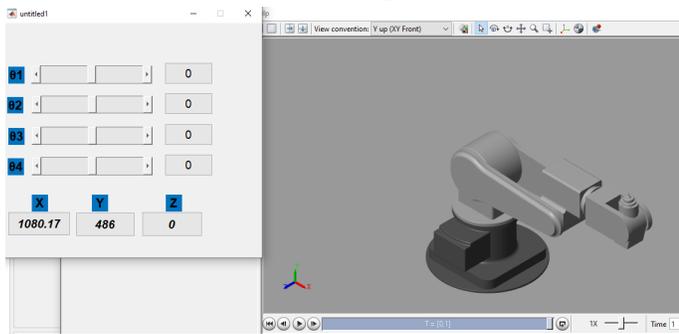


Fig. 9 Captura de la posición por defecto del brazo robótico

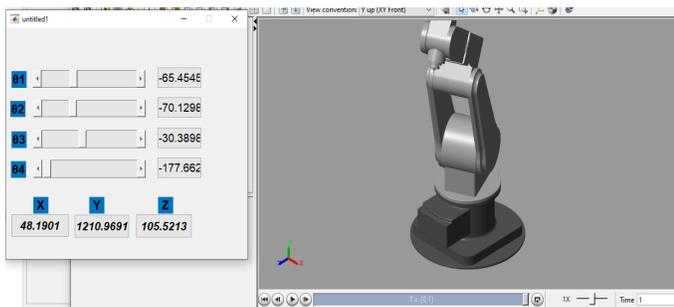


Fig. 10 Captura de la posición del brazo robótico con ángulos distintos.

- En la Programación de movimientos del brazo robótico en Arduino

#### Declaración de Librerías utilizadas

A continuación se describe las librerías usadas en la programación del brazo robótico en Arduino:

Servo.h: Permite a las placas Arduino controlar diversas variedades de servomotores. Esta biblioteca permite controlar una gran cantidad de servos.

#### Descripción de propiedades y funciones:

En la codificación de los servos, se hallan funciones tipo “attach” y “write” las cuales son especificadas a continuación mediante un ejemplo, en este sentido, luego de haber declarado el objeto, por ejemplo un servo “servomotor 1.” al interior del “Setup”, se encuentra un “servomotor 1.attach”; esta propiedad “attach” contiene los parámetros de afinación del servomotor y principalmente para direccionar el objeto con el pin de conexión por el cual recibirá la señal digital.

La propiedad “Objeto.attach” incorpora el pin de conexión, el valor mínimo, el valor máximo // , en los cuales los valores máximo y mínimo depende del tipo de servomotor. A continuación extracto de código de programación de brazo robótico donde se muestra la propiedad .attach ()

```
void setup () [
servomotor1.write(11) ;
servomotor1.write(10) ;
servomotor1.write (5);
Dentro del loop, hallaremos servo motor 1.write(0);
```

void loop () (

A continuación extracto de código de programación de brazo robótico donde se muestra la propiedad .write() //Reinicio

```
servomotor1.write(0);
servomotor1.write(45) ;
servomotor1.write (50) ;
```

La propiedad “write” especifica el número de grados en el que se quiere inicializar el servomotor, en la mayoría de los casos de servomotores se considera el rango de 0 a 180.

#### Respecto a la propiedad despliegues y movimiento:

La siguiente codificación es para obtener el despliegue de cada articulación que incluye el brazo robótico. El caso de considerar centrado básicamente en la iteración de ángulos con rango de -90 a 90 grados, para mover cada servomotor e impulsar el funcionamiento de la pinza, esto sirve también para aplicar a cualquier articulación del brazo.

Como consideración en la programación basado en arduino, por la guía o por la hoja de datos ( datasheet), se tiene que emplear un retardo (delay (20)) es decir 20 milisegundos como mínimo entre una orden y la orden consecutiva de movimiento, de ese modo podremos garantizar un óptimo funcionamiento del servomotor al recibir los cambios de ángulos que se indique a los servos.

Para declarar e indicar la iteración del movimiento se emplea la función “For”, cuya condicional permite declarar y

ejecutar el bucle, es decir una asignación a la variable de control del bucle de un valor inicial, el cual puede ser representado con un incremento y decremento que se encarga de modificar la variable de control y la expresión lógica que es la que puede determinar el fin del bucle.

3) Para el Coppeliasim y basado en los conceptos teóricos :

Como el Coppeliasim, es un programa conocido como V-REP para simular robots, en esta etapa del desarrollo se menciona que el programa dará soporte de apoyo al Software Arduino 1 debido a que este último no proporciona simulación visual para realizar la comparación con el Matlab, es necesario usar este software de apoyo.

Para apreciar algunas ventajas es necesario mencionar que presenta controles de uso, rotación de cámara, jerarquía de escena y simulación con motores físicos como principales componentes.

El lenguaje de programación que se maneja es el lenguaje Lua, que es un lenguaje Script ligero y eficiente, se constituye en un soporte procedimental, que se orienta a objetos, gestión y descripción de datos. Se implementa como una biblioteca escrita en C ++. A continuación extracto de código de programación de brazo robótico donde se muestra la propiedad .attach.

```
//Coger objetos
for (int i=0 ; i<=45; i++){
servomotor3.write(i);
delay(25);
}
delay (1000);
for (int i=0 ; i<=90; i++){
servomotor2.write(i);
delay(25);
}
delay (1000);
for (int i=0 ; i<=90; i++)
servomotor1.write(i);
delay(25);
}
delay (1000);
for (int i=90 ; i<=0: i--)[
servomotor3.write(i);
delay (25);
]
delay (1000);
for (int i=90 ; i<=0: i--)[
servomotor1.write(i);
delay (25);
]
delay (1000);
for (int i=90 ; i<=0: i--)[
servomotor2.write(i);
delay (25);
]
}
```

delay (1000);

A continuación se presenta la aplicación en un conjunto de pasos:

Paso 1:

Se exploran los controles de interfaz básica para desplazar la cámara, mediante el mouse.

Controles de movimiento

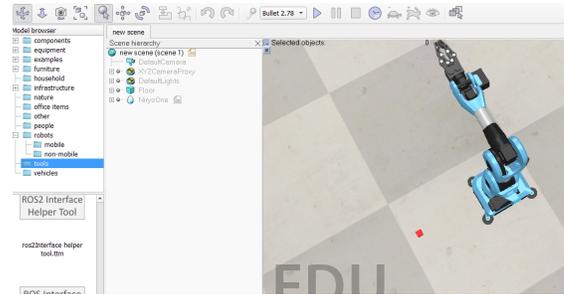


Fig. 11 Captura de los controles de movimiento

Controles de traslado.

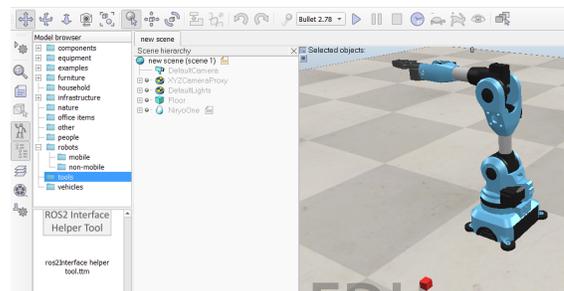


Fig. 12 Captura de los controles de movimiento  
Controles de traslado

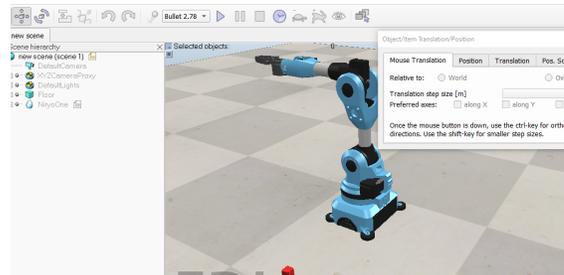


Fig. 13 Captura de los controles de desplazamiento

Paso 2:

Dentro de las funciones del lenguaje de programación Lua se presenta algunas de las principales funciones de programación:

TABLA II  
FUNCIONES DEL LENGUAJE DE PROGRAMACIÓN

and(para definir condiciones lógicas)	return(para regresar al bucle)	local(para revisar una variable local)
nil (para definir valores nulos )	end(condiciones lógicas para culminar de funciones)	not(definir función lógica de negada)
funcion(para definir una función)	break(secuencia de pausa)	true(para definir función lógica de falsedad)

TABLA III

OPERADORES MATEMATICOS

Operadores	Descripción	Ejemplo
+	Adición de dos variables	A+B
-	Diferencia de dos variables	A-B
*	Multiplicación de variables	A*B
/	Divide al módulo de variable	A/B

TABLA IV  
OPERACIONES DE RELACIÓN

Operadores	Descripción	Ejemplo
==	Comparamos la funciones si dos números son iguales	A==B
> o >=	Vemos si dos números son mayores o iguales	A > o >= B
< o <=	para revisar si dos números son menores o iguales	A < o <= B

En relación a las declaraciones que se utiliza:

Se tiene la cadena de texto tipo String, para concatenar dos cadenas de texto.

Para la declaración función, acepta dos argumentos con variables numéricas VAR1 y para la llamada de salida utiliza RESULT a la función.

Bajo la condición SI nos devolverá la asignación de dos argumentos de salida, previamente se tiene la declaración de dos argumentos.

Se forma la función sum(a,b) para declarar dos variables estas pueden tener dos entradas como 2 y 3 declarando suma próximamente los tres ejes, que luego será acompañado por el bucle for.

```
funcion sum(a,b)
return a+b
end
var1=2
var2=3
result=sum(var1,var2)
```

Fig. 14 Código de programación de variable asignadas.

En relación a la figura 13 para la salida de la variable A[i] de contenido nulo, iniciará una fila vacía dada por A[i][j], luego se le asigna dos elementos 1 y 0 para iniciar los procesos o leer los resultados de parada.

```
A={}
for i=1,3,1 do
A[i]={}
for j=1,3,1 do
if i==j then
A[i][j]=1
else
A[i][j]=0
end
end
end
```

Fig. 15 Código de formación de la salida de la variable A[i].

Paso 3:

Descripción para la ejecución en Coppelia Sim en Script para la introducción del lenguaje Lua señalado para la herramienta TOOLS presente en el Coppelia Sim.

El código presentado está ya establecido, más no desarrollado por el equipo, para el tipo de brazo robótico NIRYO ONE dentro del Script.

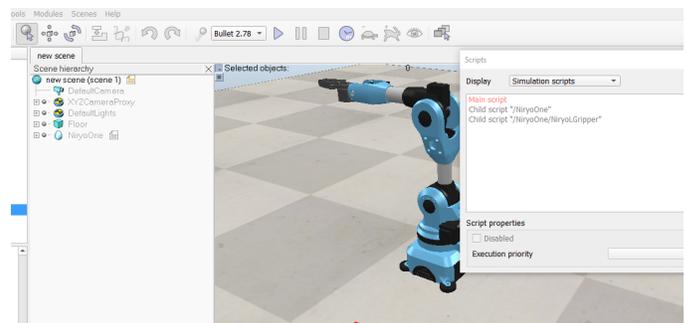


Fig. 16 Interfaz gráfica del brazo robótico-CoppeliaSim

El tiempo de ejecución será en cuestión de milisegundos, la secuencia dada estará en un subproceso que reanudará la secuencia de rutina "sim.setThreadSwitchTiming o sim.set Thread Automatic Switch.", esto se dará en un siguiente paso en la simulación. Con los comandos se dará una secuencia sincronizada para un bucle de simulación principal.

```
function sysCall_actuation()
if coroutine.status(corout) ~= 'dead' then
local ok,errorMsg=coroutine.resume(corout)
if errorMsg then
error(debug.traceback(corout,errorMsg),2)
end
end
end

function movCallback(config,vel,accel,handles)
for i=1,#handles,1 do
if sim.isDynamicallyEnabled(handles[i]) then
sim.setJointTargetPosition(handles[i],config[i])
else
sim.setJointPosition(handles[i],config[i])
end
end
end
```

Fig. 17 Secuencia de código en CoppeliaSim

Configuración de los vectores para los comandos de rutina secuenciados de manera lineal.

```
function coroutineMain()
    local jointHandles={}
    for i=1,6,1 do
        jointHandles[i]=sim.getObject('./Joint',{index=i-1}
    end

    local connection=sim.getObject('./connection')
    local gripper=sim.getObjectChild(connection,0)
    local gripperName="NiryonoGripper"
    if gripper~-1 then
        gripperName=sim.getObjectAlias(gripper,4)
    end
end
```

Fig. 18 Continuación de secuencia de comando en CoppeliaSim

Este proceso brinda libertad a los comandos en un lenguaje Script de Lua, cabe señalar que se puede usar un Script de lenguaje Python.

#### IV. RESULTADOS

La cantidad de líneas de código que se usó en el programa Matlab es de 522 y por otro lado, en Arduino se usaron 75 líneas de código.

Ambos códigos son accesibles por el estudiante que experimenta y en ese sentido dependiendo de su nivel de preparación y conocimiento puede hacer modificaciones que generen diferentes experiencias.

La operación de la experiencia desarrollada en Matlab es sencilla usando la interfaz gráfica de usuario, la modificación de otras variantes de la experiencia requiere conocimiento especial.

Se verifica que la asociación Arduino-CoppeliaSim simplifica el control del brazo robótico y lo hace amigable para la operación. La modificación del código de Arduino es mucho más fácil que la modificación del código de Matlab

El resultado comparativo se presenta en la tabla V:

TABLA V  
CUADRO COMPARATIVO

Desarrollo en Matlab	Desarrollo en Arduino y Coppelia simu
Desarrollo y aprendizaje teórico para la implementación de códigos usados basados en señales.	Códigos ya preestablecidos en algunos casos y en otros se debe desarrollar de forma libre.

No existe una gama de robots ambientados a los requerimientos del sistema a implementar	Existen algunos robots que se pueden usar ya desarrollados.
Se pueden generar secuencias visuales mediante vectores y movimientos por medio de códigos.	Se puede desarrollar una secuencia de códigos y elaborar imágenes en tres dimensiones de forma detallada.
Resulta ser algo más trabajoso el uso para la enseñanza	Resulta ser más práctico si en caso se desea modelar un robot para la solución de una situación.

A continuación se presentan los diagramas de barras elaborados en base a encuestas aplicadas a una muestra de estudiantes de la Facultad de Ingeniería Electrónica y Eléctrica, que se utiliza como índice para verificar la facilidad de aplicación de los programas en Arduino y Matlab por parte de los estudiantes. Las siguientes gráficas muestran el porcentaje del nivel de satisfacción para el uso comparativo.



Fig. 19 Resultados de la encuesta de satisfacción en porcentaje de los usuarios con Matlab.



Fig. 20 Resultados de la encuesta de satisfacción en porcentaje de los usuarios de Arduino.

## V. CONCLUSIONES Y OBSERVACIONES

1. Se logró desarrollar y operar a través de los software Matlab y Arduino los modelos de robots seleccionados en las herramientas Coppelia Sim y Solidworks.
2. Es posible diseñar y experimentar con la robótica sin necesidad de utilizar ensamblaje y profundizar en la robótica, de ese modo se logra concluir que la enseñanza de los softwares Arduino y Matlab puede ser método de enseñanza para estudiantes de primeros niveles académicos universitarios de la escuela de Ingeniería electrónica y eléctrica.
3. Entre los principales beneficios que se considera en Arduino al igual que Matlab es que los diseños tienen la posibilidad de ser mejorados, haciendo variaciones en la codificación programable y poder implementar la cinemática inversa o incluso sensores ultrasonidos para que el movimiento del control pueda hacer reconocimiento de las tareas asignadas repetitivas en el movimiento del sistema robótico.
4. Relativamente en las universidades nacionales hallaremos limitaciones económicas, aquello se puede inferir de la ausencia de equipos y herramientas tecnológicas en los laboratorios pese a esa problematización, el presente artículo trata de forjar un motivante en los estudiantes de primeros niveles académicos.

## REFERENCIAS

- [1] Pérez Díaz, A. Macías Lazcano, M. Vázquez Chávez, E. Construcción de un prototipo de brazo robótico manipulado con Arduino I. 2017.
- [2] Ibañez Mariñelarena, A., Andueza Unanua, A. Programación de un brazo robótico basado en Arduino y controlado remotamente. Trabajo de fin de grado. Ingeniería Industrial, Informática y de Telecomunicación, UPNA. Pamplona, Navarra. España. 2018.
- [3] Barker, B.S., & Ansoorge, J. (2007). Robotics as means to increase achievement scores in an informal learning environment. *Journal of Research on Technology in Education*, 39(3), 229-243.

- [4] Eranki, VKP; Reddy, Gurudu, R. Diseño y Análisis Estructural de un Brazo Robótico. 2017
- [5] Lopez Barron, E. (2022). Generación de trayectorias caóticas en pequeños robots móviles. Centro de Investigación Científica y de Educación Superior de Ensenada, Baja California. [https://cicese.repositorioinstitucional.mx/jspui/bitstream/1007/3706/3/tesis\\_Eduardo%20Lopez%20Barron\\_21%20abril%202022.pdf](https://cicese.repositorioinstitucional.mx/jspui/bitstream/1007/3706/3/tesis_Eduardo%20Lopez%20Barron_21%20abril%202022.pdf)
- [6] Burbano Merino, J. C., & Cárdenas León, L. A. (2022, 20 marzo). DESARROLLO DE UN ROBOT MÓVIL CON SISTEMA DE RECOLECCIÓN DE OBJETOS CONTROLADO REMOTAMENTE PARA FINES DIDÁCTICOS. <https://dspace.ups.edu.ec/handle/123456789/24008>
- [7] Markley Erazo, J. (2020, octubre). SIMULACIÓN DE UN ROBOT CON 6 GRADOS DE LIBERTAD UTILIZANDO TOOLBOX ROBOTICS DEL SOFTWARE MATLAB. ESCUELA POLITÉCNICA NACIONAL. <https://bibdigital.epn.edu.ec/bitstream/15000/21243/1/CD%2010760.pdf>
- [8] Vicente González, P. (2022, julio). Diseño mecánico de un robot SCARA paralelo 3RRR. UNIVERSIDAD DE VALLADOLID ESCUELA DE INGENIERÍAS INDUSTRIALES. <https://uvadoc.uva.es/bitstream/handle/10324/54191/TFG-I-2244.pdf?sequence=1&isAllowed=y>