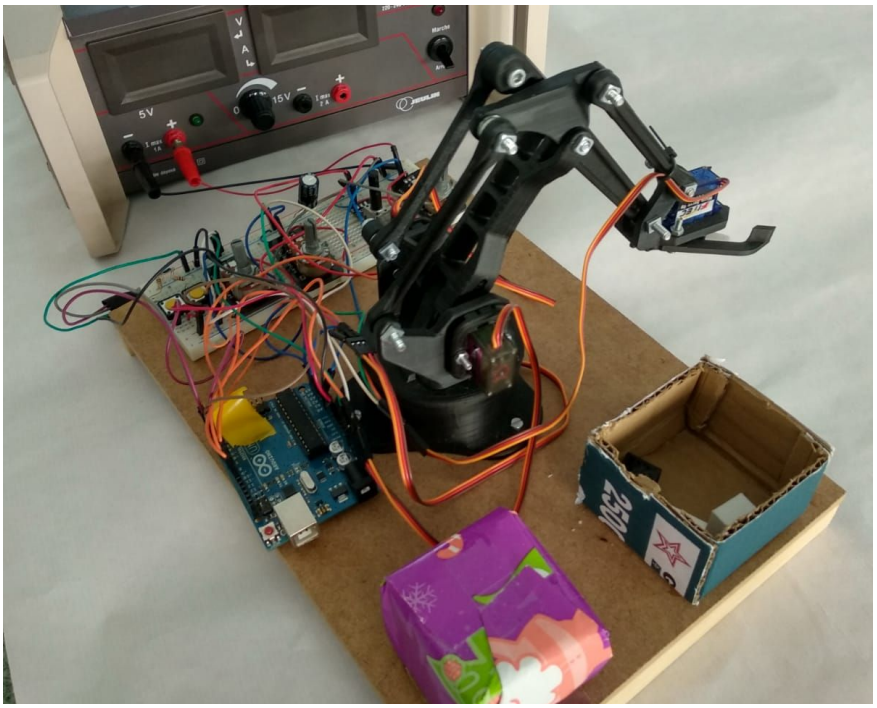
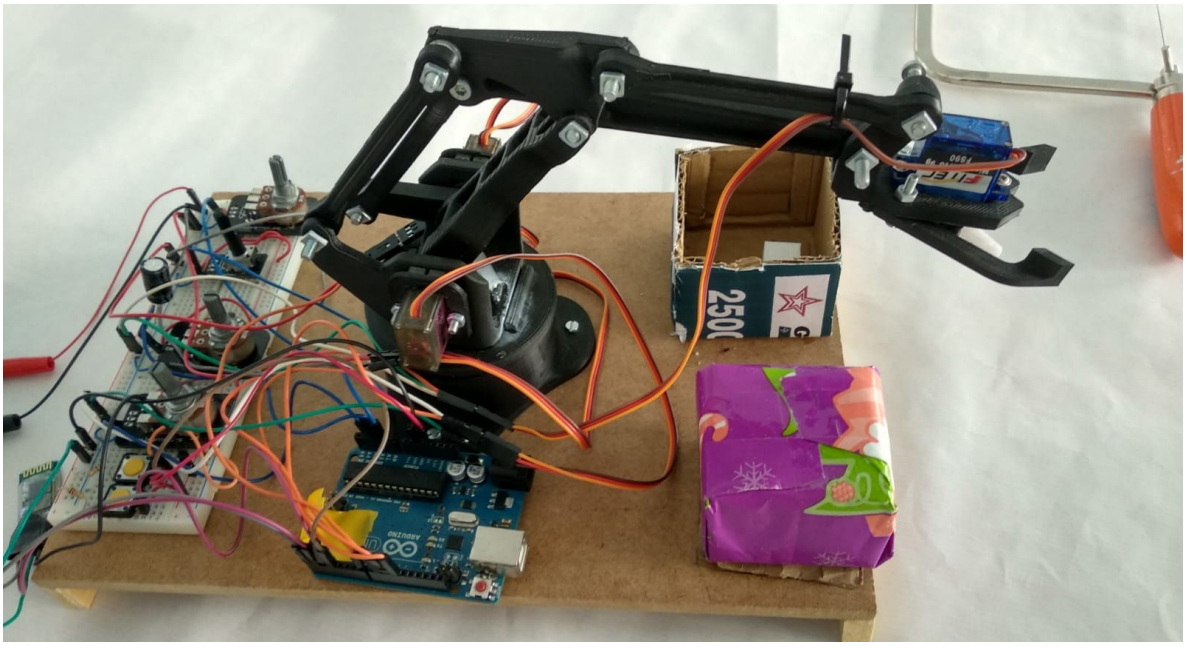


MEMORIA FINAL PROYECTO



Máximo Romero Martínez
Juan José Nogales Herrera
Gonzalo Romero González
Miguel Cruz Real

ÍNDICE

1.- Finalidad de nuestro sistema

La finalidad de nuestro proyecto es conseguir un brazo robótico (creado por impresión 3D y ensamblado por nosotros) que sea capaz de transportar objetos no muy pesados de un lugar a otro y que pueda ser controlado al gusto del usuario.

Nuestro Producto Mínimo Viable (PMV) era poder mover el brazo con unos potenciómetros. Cada potenciómetro estaba destinado a controlar un servomotor diferente del brazo.

Con el tiempo , al lograr el PMV, continuamos ampliando y mejorando el brazo. Por ejemplo introducimos un movimiento paulatino, que dotaba al brazo de movimiento fluido y suave. También creamos una aplicación con App Inventor, que nos permitía controlar al brazo desde el móvil gracias a un módulo Bluetooth que acoplamos al hardware del proyecto.

2.- Búsqueda de información

Como en cualquier proyecto de una mínima extensión, hemos precisado de multitud de información para acabarlo.
Esta información se ha basado en fotos , vídeos, instructables, páginas web , y básicamente cualquier formato que pudiese suponer algún tipo de ayuda para el proyecto, que ahora detallo en más profundidad con los siguientes enlaces.

Modelos de brazos robóticos libres

- Brazo mecánico MK1 by daGHIZmo
- Brazo mecánico MK2 by daGHIZmo
- Pedro Robot

Resolver robots articulados con Arduino
https://www.luisllamas.es/resolver-robots-articulados-con-arduino/

Recopilación de brazos robóticos;
https://tecnoloxia.org/robotica/brazo-robot/

Instrucciones para ensamblar las piezas del brazo:
https://www.instructables.com/id/EEZYbotARM/

Brazo Robótico controlado con Arduino a través de una app.
https://arduino.blog/2017/01/01/controla-un-brazo-robotico-con-arduino-y-android/

Video mk1 explicado
https://youtu.be/r0965C587mE

Placas usadas (placa pololu)
https://www.pololu.com/product/1352
https://www.pololu.com/product/1350

Control con app inventor
https://www.instructables.com/id/Android-APP-to-Control-a-3DPrinted-Robot/

Shield sensor arduino
https://www.iberobotics.com/producto/arduino-sensor-shield/

Shields para servos
https://www.adafruit.com/product/1411
https://www.prometec.net/motorshield-servos/
https://www.openimpulse.com/blog/products-page/product-category/16-channel-12-bit-pwm-servo-shield-arduino/

Información sobre potencia externa con servo
https://docs.google.com/presentation/d/10si1zTxGDYMNOWSzN9eRyCKBWn-TUtYLX_fksx0kEyE/edit#slide=id.g30dfc9d692_0_42

Data sheet servomotores SG90 (azules)
http://www.ee.ic.ac.uk/pcheung/teaching/de1_ee/stores/sg90_datasheet.pdf

Data sheet servomotores MG90S (negros)
https://engineering.tamu.edu/media/4247823/ds-servo-mg90s.pdf

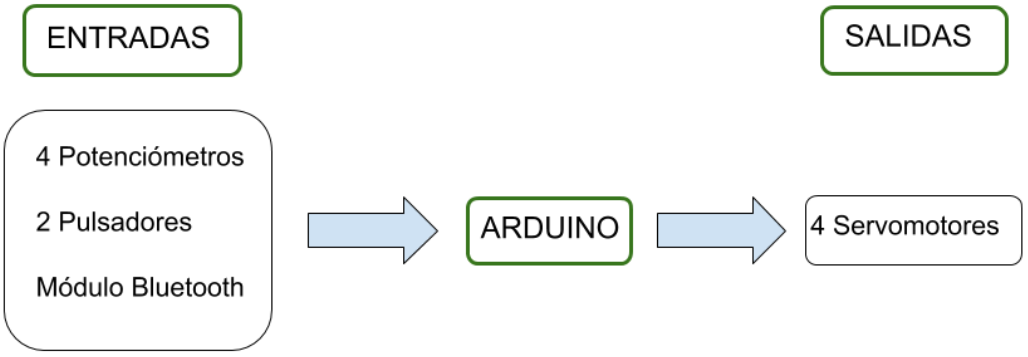
Salidas analógicas Luis Llamas
https://www.luisllamas.es/salidas-analogicas-pwm-en-arduino/

Código App Inventor
https://www.youtube.com/watch?v=OO3RmjF_KT8

Interrupciones
https://www.luisllamas.es/que-son-y-como-usar-interrupciones-en-arduino/
https://www.luisllamas.es/debounce-interrupciones-arduino/
https://programarfacil.com/blog/arduino-blog/interrupciones-con-arduino-ejemplo-practico/

3.- Hardware

- Esquema de entradas y salidas

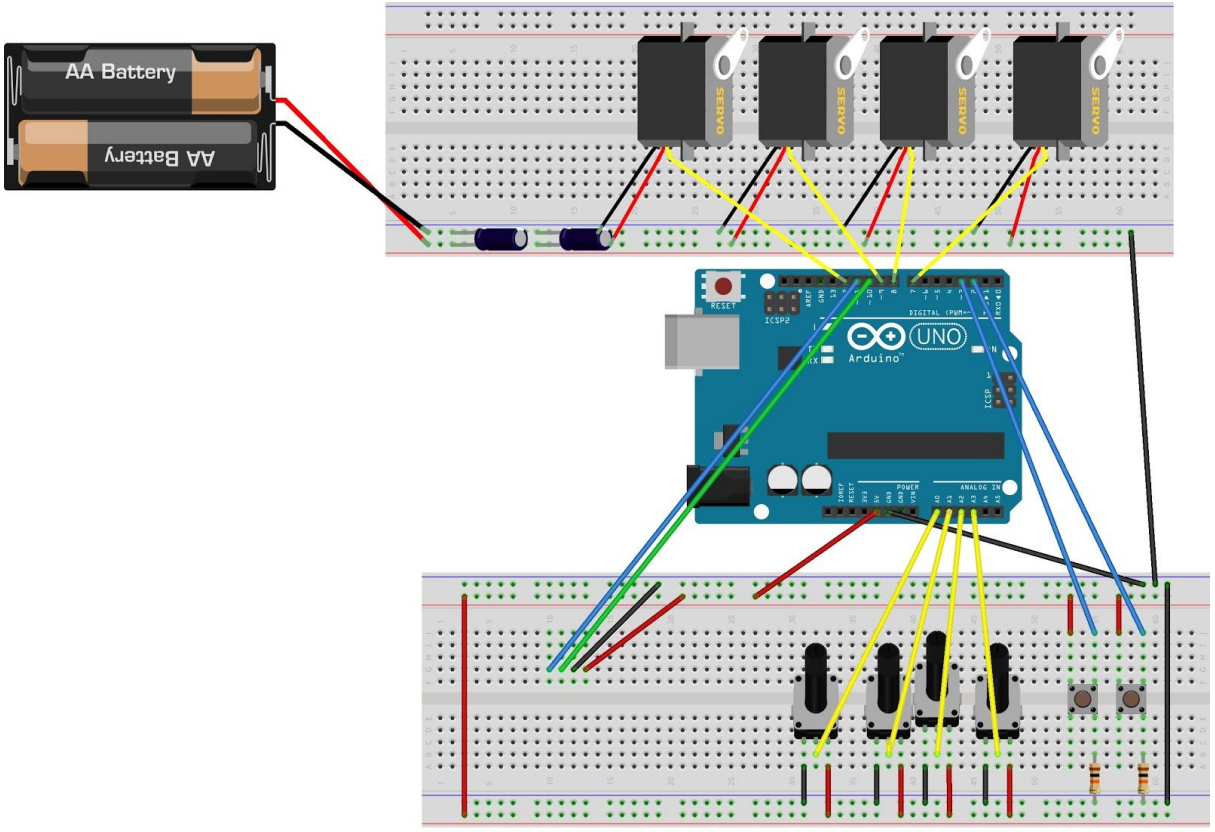


- Lista de materiales

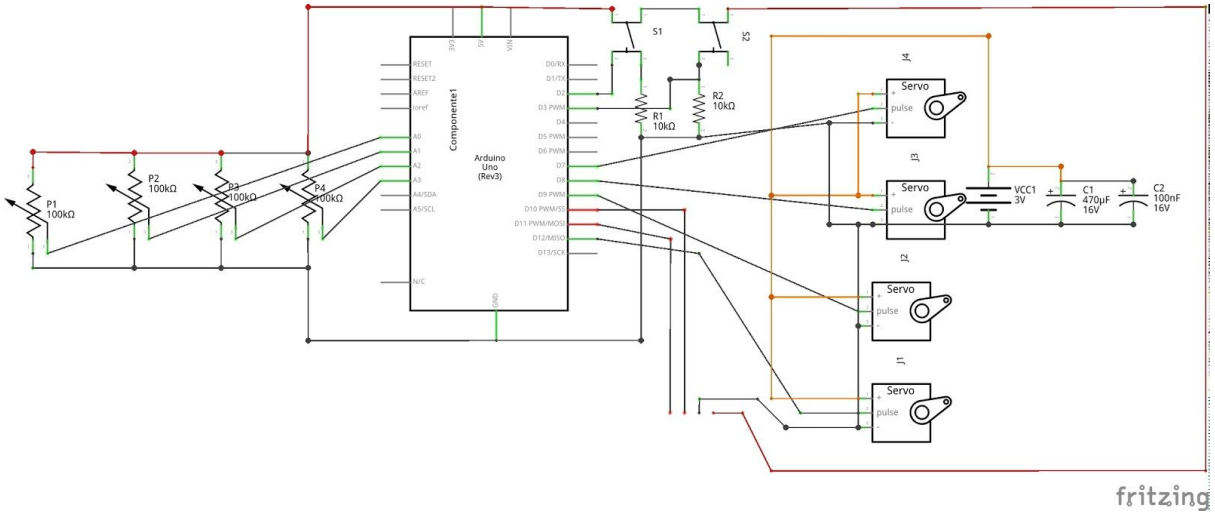
TORNILLERIA		
Nº	**Descripción**	**Cantidad**
1	Caja	1
2	M4 tuerca autoblocante	7
3	M4 arandela plana	15
4	M3 tuerca hexagonal	7
5	M3 x 30mm tornillo	1
6	M3 arandela plana	2
7	M3 x 12mm tornillo hexagonal	4
8	M3 x 12mm tornillo cabeza estrella	2
9	M3 x 20mm tornillo cabeza estrella	2
10	M4 x 20mm tornillo cabeza hexagonal	5
11	Tubo de cobre 4 x 3 x 22 + n°1 4 x 3 x 26	1
PIEZAS DE IMPRESIÓN 3D		
Nº	**Descripción**	**Cantidad**
1	n°1 EBA_01.00.001.STL	1
2	EBA_01.00.002_vertical_drive_arm.STL	1
3	· n°3 EBA_01.00.003_link.STL	2
4	n°1 EBA_01.00.004_forward_drive_arm.STL	1
5	· n°1 EBA_01.00.005_horizontal_arm.STL	1

6	· n°1 EBA_01.00.006_triangular_link.STL	1
7	· n°2 EBA_01.00.007_servo_plate.STL	1
8	· n°1 EBA_01.00.08_basement.STL	1
9	n°1 EBA_01.00.09_round_plate.STL	2
10	· n°1 EBA_01.00.010_R01_claw_support.STL	1
11	· n°1 EBA_01.00.011_R01_right_finger.STL	1
12	· n°1 EBA_01.00.012_R01_left_finger.STL	1
13	· n°1 EBA_01.00.013_drive_gear.STL	1
14	· n°1 EBA_01.00.014_R01_driven_gear.STL	2
ELECTRÓNICA		
Nº	**Descripción**	**Cantidad**
1	Tower Pro MG90S servos	3
2	SG90 servo para la pinza	1
3	Pulsadores	2
4	Potenciómetros	4
5	Cableado Macho-Macho	32
6	Cableado Macho-Hembra	4
7	Condensador	1
8	Placa de Arduino 1	1
9	Módulo Bluetooth	1
10	Resistencias 10000 Ohmios	2
11	Fuente de Alimentación Externa de 5V	1
12	Protoboard	1

- Esquema de la protoboard



- Esquema electrónico



4.- Software

- Código App Inventor

```
#include <Servo.h>

Servo servo1;   // se declara la variable myservo1
Servo servo2;  // se declara la variable myservo2
Servo servo3;  // se declara la variable myservo3
Servo servo4;  // se declara la variable myservo4
char a;
String readString;

void setup() {

  pinMode(13,OUTPUT); // se declara el pin 13 como salida

  servo1.attach(7); // se adjunta la variable myservo1 al pin 7
  servo2.attach(9); // se adjunta la variable myservo2 al pin 9
  servo3.attach(10);  // se adjunta la variable myservo3 al pin 10
  servo4.attach(11); // se adjunta la variable myservo4 al pin 11

  Serial.begin(9600);

servo1.write(8); // se indican las variables a escribir en mysevo1
servo2.write(100); // se indican las variables a escribir en mysevo2
servo3.write(164); // se indican las variables a escribir en mysevo3
servo4.write(90); // se indican las variables a escribir en mysevo4

delay(10);
}

void loop() {

 if (Serial.available()) {
    a = Serial.read(); // variable a leer

    if(a=='A'){ // si a es igual a A , hablamos del motor 1
      motor1();
    }

    if(a=='B'){ // si a es igual a B , hablamos del motor 2
      motor2();
    }
```

```
    if(a=='C'){ // si a es igual a C , hablamos del motor 3
      motor3();
    }

     if(a=='D'){ // si a es igual a D , hablamos del motor 4
      motor4();
    }
     if(a=='E'){ // si a es igual a E, se activa el sistema
     digitalWrite(13,HIGH); // se activa el pin 13
     delay(10); // espera de 1 ms
    }
      if(a=='F'){  // si a es igual a F, se apaga el sistema
     digitalWrite(13,LOW);  // se apaga el pin 13
     delay(10); // espera de 1 ms
    }
  }
}

void motor1(){
        delay(10); // espera de 1 ms
        while (Serial.available()) {

          char b = Serial.read();
          readString += b;
        }
        if (readString.length() >0) {
          Serial.println(readString.toInt());
          servo1.write(readString.toInt());
          readString=""; // Clear string
        }
}

void motor2(){
        delay(10); // espera de 1 ms
        while (Serial.available()) {
          char b = Serial.read();
          readString += b;
        }
        if (readString.length() >0) {
          Serial.println(readString.toInt());
          servo2.write(readString.toInt());
          readString="";
        }
}

void motor3(){
        delay(10);
```

```
        while (Serial.available()) {
          char b = Serial.read();
          readString += b;
        }
        if (readString.length() >0) {
          Serial.println(readString.toInt());
          servo3.write(readString.toInt());
          readString="";
        }
}
void motor4(){
        delay(10);
        while (Serial.available()) {
          char b = Serial.read();
          readString += b;
        }
        if (readString.length() >0) {
          Serial.println(readString.toInt());
          servo4.write(readString.toInt());
          readString="";
        }
}
```

- Código manual / automático

```
#include <Servo.h> //incluimos la librería servo

const int buttonPin1 = 2;  // pines de botón 1
const int buttonPin2 = 3;  // pines de botón 1

// Variables will change:
int volatile autom = 0;   //variable movimiento automático
int volatile posicion = 0;// variable posición
int buttonState1;        // the current reading from the input pin
int lastButtonState1 = LOW;   // the previous reading from the input pin
int buttonState2;       // the current reading from the input pin
int lastButtonState2 = LOW; // the previous reading from the input pin

unsigned long lastDebounceTime1 = 0;  // the last time the output pin was toggled
```

```
unsigned long debounceDelay1 = 50; // the debounce time; increase if the output flickers
unsigned long lastDebounceTime2 = 0;  // the last time the output pin was toggled
unsigned long debounceDelay2 = 50; // the debounce time; increase if the output flickers

// declaramos 4 objetos servomotores
Servo myservoBase;
Servo myservoArriba;
Servo myservoPinza;
Servo myservoAtras;

// declaramos las variables para los pines de los 4 servomotores
const int servoPinBase = 7;
const int servoPinArriba = 9;
const int servoPinPinza = 10;
const int servoPinAtras = 11;

// declaramos las variables para los pines de los 4 potenciómetros
const int potPinBase = A0;
const int potPinArriba = A1;
const int potPinPinza = A2;
const int potPinAtras = A3;

// declaramos las variables de los valores de cada potenciómetro
int potValueBase = 0;
int potValueArriba = 0;
int potValuePinza = 0;
int potValueAtras = 0;

//declaramos las variables de los valores de los angulos de cada servo
int angleBase=0;
int angleArriba=0;
int anglePinza=0;
int angleAtras=0;


void setup() {


  Serial.begin(9600);// establecemos la velocidad de conexión con el puerto serie

  // asociamos cada objeto servo a su pin correspondiente
  myservoBase.attach (servoPinBase);
  myservoArriba.attach (servoPinArriba);
  myservoPinza.attach (servoPinPinza);
  myservoAtras.attach (servoPinAtras);
  //establecemos una posición inicial del brazo
  myservoBase.write (50);
  myservoArriba.write (101);
  myservoPinza.write (167);
  myservoAtras.write (32);
```

```
}
void loop() {

  debounce1();//leemos el pulsador 1

  debounce2();//leemos el pulsador 2

  if (autom == 1) {
  if (posicion == 0){
  posicion0();//si está en posición 0 lo llevamos a la posición inicial
  }
  else {
  automatico();//si está en modo automático y posición 1 (en movimiento) hace el
movimiento automático
  }
  }
  else {
  if (posicion == 0){
  posicion0();//si está en posición 0 lo llevamos a la posición inicial
  }
  else {
  manual();//si está en modo manual (automático=0) y posición 1 (en movimiento) hace el
movimiento manual
  }
  }

}
void debounce1() {
  int reading = digitalRead(buttonPin1);//leemos el botón y guardamos el valor en la
variable

  if (reading != lastButtonState1) {
  // reset the debouncing timer
  lastDebounceTime1 = millis();
  }

  if ((millis() - lastDebounceTime1) > debounceDelay1) {
  // whatever the reading is at, it's been there for longer than the debounce
  // delay, so take it as the actual current state:

  // if the button state has changed:
  if (reading != buttonState1) {
    buttonState1 = reading;

    //solo cambia el valor de la variable automático si el borón está en estado HIGH
    if (buttonState1 == HIGH) {
      autom = !autom;
    }
```

```
  }
  }


  lastButtonState1 = reading;//actualizamos el último valor del estado del botón
  //imprimimos el modo y posicion en la que nos encontramos para que el usuario lo
conozca
  Serial.print ("Modo");
  Serial.println (autom);
  Serial.print ("Posicion");
  Serial.println (posicion);
}

void debounce2() {
  int reading = digitalRead(buttonPin2);//leemos el botón y guardamos el valor en la
variable

  if (reading != lastButtonState2) {
  // reset the debouncing timer
  lastDebounceTime2 = millis();
  }

  if ((millis() - lastDebounceTime2) > debounceDelay2) {
  // whatever the reading is at, it's been there for longer than the debounce
  // delay, so take it as the actual current state:

  // if the button state has changed:
  if (reading != buttonState2) {
    buttonState2 = reading;

   //solo cambia el valor de la variable automático si el borón está en estado HIGH
    if (buttonState2 == HIGH) {
      posicion = !posicion;
    }
  }
  }
  lastButtonState2 = reading;//actualizamos el último valor del estado del botón
  //imprimimos el modo y posicion en la que nos encontramos para que el usuario lo
conozca
  Serial.print ("Modo");
  Serial.println (autom);
  Serial.print ("Posicion");
  Serial.println (posicion);
}
void automatico(){

  //se realiza siempre la misma rutina de movimiento de los 4 servo. Se usan funciones
que envían y reciben valores.
  paulatinoBase (50,22);
  paulatinoArriba (101,10);
```

```
  paulatinoAtras (32,129);
  myservoPinza.write (16);

  paulatinoBase(22, 68);
  paulatinoArriba (10,162);
  paulatinoAtras (129,14);
  myservoPinza.write (160);

  paulatinoBase(68, 50);
  paulatinoArriba (162,101);
  paulatinoAtras (14,32);
  myservoPinza.write (13);

  posicion=0;//se actualiza el valor de la variable posición para que vuelva a la posición inicial cuando termine la rutina

}

void paulatinoBase (int angle_antBase, int angleBase) { //se encarga del movimiento paulatino de la base

  /*si el ángulo anterior es más pequeño se van a ir sumando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  if (angle_antBase < angleBase) {
  for (int i = angle_antBase; i <= angleBase; i++) {
  myservoBase.write (i);
  delay(20);
  }
  }
  /*si el ángulo anterior es más grande se van a ir restando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  else {
  for (int i = angle_antBase; i >= angleBase; i--) {
  myservoBase.write (i);
  delay(20);
  }
  }
}

void paulatinoAtras (int angle_antAtras, int angleAtras) { //se encarga del movimiento paulatino del servo que mueve el brazo alante-atrás

 /*si el ángulo anterior es más pequeño se van a ir sumando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  if (angle_antAtras < angleAtras) {
  for (int i = angle_antAtras; i <= angleAtras; i++) {
  myservoAtras.write (i);
```

```
  delay(20);
  }
  }
  /*si el ángulo anterior es más grande se van a ir restando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  else {
  for (int i = angle_antAtras; i >= angleAtras; i--) {
  myservoAtras.write (i);
  delay(20);
  }
  }
}

void paulatinoArriba (int angle_antArriba, int angleArriba) { //se encarga del movimiento paulatino del servo que mueve el brazo arriba-abajo

  /*si el ángulo anterior es más paqueño se van a ir sumando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  if (angle_antArriba < angleArriba) {
  for (int i = angle_antArriba; i <= angleArriba; i++) {
  myservoArriba.write (i);
  delay(20);
  }
  }
  /*si el ángulo anterior es más grande se van a ir restando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  else {
  for (int i = angle_antArriba; i >= angleArriba; i--) {
  myservoArriba.write (i);
  delay(20);
  }
  }
}

void manual(){ //usamos una función diferente para cada servomotor
  ServoBase ();
  ServoArriba ();
  ServoPinza ();
  ServoAtras ();

}

void ServoBase () {

  potValueBase = analogRead (potPinBase);//leemos el pin del portenciómetro y lo guardamos en la variable
  angleBase = map (potValueBase,0,1023,180,0);//escalamos el valor de la variable para
```

```
que quede entre 0 y 180
  myservoBase.write (angleBase);//hacemos que el servo se mueva al valor de la variable
escalada
  delay(30);
 }


void ServoArriba () {

  potValueArriba = analogRead (potPinArriba);//leemos el pin del portenciómetro y lo
guardamos en la variable
  angleArriba = map (potValueArriba,0,1023,180,0);//escalamos el valor de la variable para
que quede entre 0 y 180
  myservoArriba.write (angleArriba);//hacemos que el servo se mueva al valor de la
variable escalada
  delay(30);
 }

void ServoPinza () {

  potValuePinza = analogRead (potPinPinza);//leemos el pin del portenciómetro y lo
guardamos en la variable
  anglePinza = map (potValuePinza,0,1023,180,0);//escalamos el valor de la variable para
que quede entre 0 y 180
  myservoPinza.write (anglePinza);//hacemos que el servo se mueva al valor de la
variable escalada
  delay(30);
 }

 void ServoAtras () {

  potValueAtras = analogRead (potPinAtras);//leemos el pin del portenciómetro y lo
guardamos en la variable
  angleAtras = map (potValueAtras,0,1023,180,0);//escalamos el valor de la variable para
que quede entre 0 y 180
  myservoAtras.write (angleAtras);//hacemos que el servo se mueva al valor de la variable
escalada
  delay(30);
 }

void posicion0 (){ //se indica los valores de los servo para la posición inicial
  myservoBase.write (50);
  myservoArriba.write (101);
  myservoPinza.write (167);
  myservoAtras.write (32);
}
#include <Servo.h> //incluimos la librería servo

const int buttonPin1 = 2;  // pines de botón 1
const int buttonPin2 = 3;  // pines de botón 1
```

```
// Variables will change:
int volatile autom = 0;   //variable movimiento automático
int volatile posicion = 0;// variable posición
int buttonState1;        // the current reading from the input pin
int lastButtonState1 = LOW;   // the previous reading from the input pin
int buttonState2;       // the current reading from the input pin
int lastButtonState2 = LOW; // the previous reading from the input pin

unsigned long lastDebounceTime1 = 0;  // the last time the output pin was toggled
unsigned long debounceDelay1 = 50; // the debounce time; increase if the output flickers
unsigned long lastDebounceTime2 = 0;  // the last time the output pin was toggled
unsigned long debounceDelay2 = 50; // the debounce time; increase if the output flickers

// declaramos 4 objetos servomotores
Servo myservoBase;
Servo myservoArriba;
Servo myservoPinza;
Servo myservoAtras;

// declaramos las variables para los pines de los 4 servomotores
const int servoPinBase = 7;
const int servoPinArriba = 9;
const int servoPinPinza = 10;
const int servoPinAtras = 11;

// declaramos las variables para los pines de los 4 potenciómetros
const int potPinBase = A0;
const int potPinArriba = A1;
const int potPinPinza = A2;
const int potPinAtras = A3;

// declaramos las variables de los valores de cada potenciómetro
int potValueBase = 0;
int potValueArriba = 0;
int potValuePinza = 0;
int potValueAtras = 0;

//declaramos las variables de los valores de los angulos de cada servo
int angleBase=0;
int angleArriba=0;
int anglePinza=0;
int angleAtras=0;


void setup() {


  Serial.begin(9600);// establecemos la velocidad de conexión con el puerto serie
```

```
  // asociamos cada objeto servo a su pin correspondiente
  myservoBase.attach (servoPinBase);
  myservoArriba.attach (servoPinArriba);
  myservoPinza.attach (servoPinPinza);
  myservoAtras.attach (servoPinAtras);
  //establecemos una posición inicial del brazo
  myservoBase.write (50);
  myservoArriba.write (101);
  myservoPinza.write (167);
  myservoAtras.write (32);


}
void loop() {

  debounce1();//leemos el pulsador 1

  debounce2();//leemos el pulsador 2

  if (autom == 1) {
  if (posicion == 0){
  posicion0();//si está en posición 0 lo llevamos a la posición inicial
  }
  else {
  automatico();//si está en modo automático y posición 1 (en movimiento) hace el
movimiento automático
  }
  }
  else {
  if (posicion == 0){
  posicion0();//si está en posición 0 lo llevamos a la posición inicial
  }
  else {
  manual();//si está en modo manual (automático=0) y posición 1 (en movimiento) hace el
movimiento manual
  }
  }

}
void debounce1() {
  int reading = digitalRead(buttonPin1);//leemos el botón y guardamos el valor en la
variable

  if (reading != lastButtonState1) {
  // reset the debouncing timer
  lastDebounceTime1 = millis();
  }

  if ((millis() - lastDebounceTime1) > debounceDelay1) {
  // whatever the reading is at, it's been there for longer than the debounce
```

```
  // delay, so take it as the actual current state:

  // if the button state has changed:
  if (reading != buttonState1) {
    buttonState1 = reading;

    //solo cambia el valor de la variable automático si el borón está en estado HIGH
    if (buttonState1 == HIGH) {
      autom = !autom;
    }
  }
  }


  lastButtonState1 = reading;//actualizamos el último valor del estado del botón
  //imprimimos el modo y posicion en la que nos encontramos para que el usuario lo
conozca
  Serial.print ("Modo");
  Serial.println (autom);
  Serial.print ("Posicion");
  Serial.println (posicion);
}

void debounce2() {
  int reading = digitalRead(buttonPin2);//leemos el botón y guardamos el valor en la
variable

  if (reading != lastButtonState2) {
  // reset the debouncing timer
  lastDebounceTime2 = millis();
  }

  if ((millis() - lastDebounceTime2) > debounceDelay2) {
  // whatever the reading is at, it's been there for longer than the debounce
  // delay, so take it as the actual current state:

  // if the button state has changed:
  if (reading != buttonState2) {
    buttonState2 = reading;

   //solo cambia el valor de la variable automático si el borón está en estado HIGH
    if (buttonState2 == HIGH) {
      posicion = !posicion;
    }
  }
  }
  lastButtonState2 = reading;//actualizamos el último valor del estado del botón
  //imprimimos el modo y posicion en la que nos encontramos para que el usuario lo
conozca
  Serial.print ("Modo");
```

```
  Serial.println (autom);
  Serial.print ("Posicion");
  Serial.println (posicion);
}
void automatico(){

  //se realiza siempre la misma rutina de movimiento de los 4 servo. Se usan funciones
que envían y reciben valores.
  paulatinoBase (50,22);
  paulatinoArriba (101,10);
  paulatinoAtras (32,129);
  myservoPinza.write (16);

  paulatinoBase(22, 68);
  paulatinoArriba (10,162);
  paulatinoAtras (129,14);
  myservoPinza.write (160);

  paulatinoBase(68, 50);
  paulatinoArriba (162,101);
  paulatinoAtras (14,32);
  myservoPinza.write (13);

  posicion=0;//se actualiza el valor de la variable posición para que vuelva a la posición
inicial cuando termine la rutina

}

void paulatinoBase (int angle_antBase, int angleBase) { //se encarga del movimiento
paulatino de la base

  /*si el ángulo anterior es más pequeño se van a ir sumando valores de 1 en 1 hasta
llegar al valor deseado
  para que se mueva el servo paulatinamente */
  if (angle_antBase < angleBase) {
  for (int i = angle_antBase; i <= angleBase; i++) {
  myservoBase.write (i);
  delay(20);
  }
  }
  /*si el ángulo anterior es más grande se van a ir restando valores de 1 en 1 hasta llegar
al valor deseado
  para que se mueva el servo paulatinamente */
  else {
  for (int i = angle_antBase; i >= angleBase; i--) {
  myservoBase.write (i);
  delay(20);
  }
  }
}
```

```
void paulatinoAtras (int angle_antAtras, int angleAtras) { //se encarga del movimiento paulatino del servo que mueve el brazo alante-atrás

 /*si el ángulo anterior es más pequeño se van a ir sumando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  if (angle_antAtras < angleAtras) {
  for (int i = angle_antAtras; i <= angleAtras; i++) {
  myservoAtras.write (i);
  delay(20);
  }
  }
  /*si el ángulo anterior es más grande se van a ir restando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  else {
  for (int i = angle_antAtras; i >= angleAtras; i--) {
  myservoAtras.write (i);
  delay(20);
  }
  }
}

void paulatinoArriba (int angle_antArriba, int angleArriba) { //se encarga del movimiento paulatino del servo que mueve el brazo arriba-abajo

  /*si el ángulo anterior es más paqueño se van a ir sumando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  if (angle_antArriba < angleArriba) {
  for (int i = angle_antArriba; i <= angleArriba; i++) {
  myservoArriba.write (i);
  delay(20);
  }
  }
  /*si el ángulo anterior es más grande se van a ir restando valores de 1 en 1 hasta llegar al valor deseado
  para que se mueva el servo paulatinamente */
  else {
  for (int i = angle_antArriba; i >= angleArriba; i--) {
  myservoArriba.write (i);
  delay(20);
  }
  }
}

void manual(){ //usamos una función diferente para cada servomotor
  ServoBase ();
  ServoArriba ();
```

```
  ServoPinza ();
  ServoAtras ();

}

void ServoBase () {

  potValueBase = analogRead (potPinBase);//leemos el pin del portenciómetro y lo
guardamos en la variable
  angleBase = map (potValueBase,0,1023,180,0);//escalamos el valor de la variable para
que quede entre 0 y 180
  myservoBase.write (angleBase);//hacemos que el servo se mueva al valor de la variable
escalada
  delay(30);
 }


void ServoArriba () {

  potValueArriba = analogRead (potPinArriba);//leemos el pin del portenciómetro y lo
guardamos en la variable
  angleArriba = map (potValueArriba,0,1023,180,0);//escalamos el valor de la variable para
que quede entre 0 y 180
  myservoArriba.write (angleArriba);//hacemos que el servo se mueva al valor de la
variable escalada
  delay(30);
 }

void ServoPinza () {

  potValuePinza = analogRead (potPinPinza);//leemos el pin del portenciómetro y lo
guardamos en la variable
  anglePinza = map (potValuePinza,0,1023,180,0);//escalamos el valor de la variable para
que quede entre 0 y 180
  myservoPinza.write (anglePinza);//hacemos que el servo se mueva al valor de la
variable escalada
  delay(30);
 }

 void ServoAtras () {

  potValueAtras = analogRead (potPinAtras);//leemos el pin del portenciómetro y lo
guardamos en la variable
  angleAtras = map (potValueAtras,0,1023,180,0);//escalamos el valor de la variable para
que quede entre 0 y 180
  myservoAtras.write (angleAtras);//hacemos que el servo se mueva al valor de la variable
escalada
  delay(30);
 }
```

```
void posicion0 (){ //se indica los valores de los servo para la posición inicial
  myservoBase.write (50);
  myservoArriba.write (101);
  myservoPinza.write (167);
  myservoAtras.write (32);
}
```

5.- Evaluación final

- Qué funciona bien y qué se puede mejorar

Se podría decir que el brazo en conjunto es un sistema bastante completo que no presenta errores demasiado graves , sin embargo , hay detalles que pueden pulirse. Por ejemplo, la limpieza del hardware podría ser mejor si hubiésemos implementado la PCB. También se podría mejorar el código de interrupciones (pulsadores) para un mejor funcionamiento del mismo. Por último, podríamos haber montado el brazo de nuevo o haberlo revisado muy minuciosamente para encontrar el motivo por el cual se mueve a veces de manera poco fluida.

- Problemas que hemos tenido y soluciones

Hemos tenido problemas con los servomotores y con su funcionamiento adecuado sobre todo al principio. Esto se debía a que la fuente de alimentación externa aportaba más tensión de la indicada, por lo que los servomotores se quemaban y no funcionaban bien.

También hemos tenido problemas con el montaje del propio brazo robótico porque había piezas que no nos encajaban bien al principio o simplemente no permitían el movimiento completo y fluido del brazo.

Además, hemos tenido varios problemas con la PCB que han provocado que la pudiéramos tener al final en nuestro proyecto. Este problema ha derivado en otros relativos al hardware, provocando un montaje poco limpio y muy propenso a fallos en determinados momentos.

Por último, nos surgieron problemas con el App Inventor y la realización de su respectiva aplicación. La aplicación se quedaba pillada en algunos momentos y esto se debía a que la tensión suministrada al módulo Bluetooth sufría picos y variaciones. Esto se podía haber solucionado con un divisor de tensión pero nos dimos cuenta ya al final del proyecto. Aún así este problema no nos privó de introducir el modo del App Inventor en nuestro brazo robótico.

- Propuestas de mejora y ampliación del proyecto

Al comienzo del proyecto , pensamos en implementar un selector de color en el brazo, pero por falta de tiempo , no vamos a poder realizarlo. Sería una propuesta de mejora muy interesante , en la que el brazo detectaría una serie de objetos de diferentes colores, los recogería y depositaría los objetos de cada color en plataformas diferentes.

También se podría realizar otro brazo más avanzado (Mk2), con más potencia en los servomotores y un mayor tamaño. Esto le permitiría un mayor rango de actuación, pudiendo levantar objetos más pesados y transportandolos más lejos.