```
/*
* ------------------------------------------------------------------------------
* Code monitors amplitude of EMG envelope, displays EMG strength on LED bar and controls
* robotic gripper by controlling servo motor.
* ------------------------------------------------------------------------------
*/

#include <Servo.h>
#define GRIPPER_STATE_BUTTON_PIN 4        //pin for button that switches defult state
                                          //of the gripper (opened/closed)
#define SERVO_PIN 2                       //pin for servo motor
#define SENSITIVITY_BUTTON_PIN 7          //pin for button that selects sesitivity
#define NUM_LED 6                         //number of LEDs in LED bar
#define GRIPPER_MINIMUM_STEP 5            //5 degree dead zone (used to avoid
                                          //aiming oscilation)
#define OPEN_MODE 1                       //default gripper state is opened
#define CLOSED_MODE 2                     //default gripper state is closed
#define MINIMUM_SERVO_UPDATE_TIME 100     //update servo position every 100ms

Servo Gripper;                            //servo for gripper
byte ledPins[] = {8, 9, 10, 11, 12, 13};  //pins for LEDs in LED bar

//EMG saturation values (when EMG reaches this value
//the gripper will be fully opened/closed)
int sensitivities[] = {200, 350, 520, 680, 840, 1000};
int lastSensitivitiesIndex = 2;           //set initial sensitivity index

int emgSaturationValue = 0;               //selected sensitivity/EMG saturation value
int analogReadings;                       //measured value for EMG
byte ledbarHeight = 0;                    //temporary variable for led bar height

unsigned long oldTime = 0;                //timestamp of last servo angle update (ms)
```

```
int oldDegrees = 0;                    //old value of angle for servo
int newDegree;                         //new value of angle for servo

unsigned long debouncerTimer = 0;          //timer for button debouncer
int gripperStateButtonValue = 0;           //variable that stores state of button
int userReleasedButton = 1;                //flag that is used to avoid multiple
                                //button events when user holds button

int currentFunctionality = OPEN_MODE;      //current default position of claw



//--------------------------------------------------------------------------------
//   Setup servo, inputs and outputs
// -------------------------------------------------------------------------------
void setup(){
  //init servo
  Gripper.attach(SERVO_PIN);

  //init button pins to input
  pinMode(GRIPPER_STATE_BUTTON_PIN, INPUT);
  pinMode(SENSITIVITY_BUTTON_PIN, INPUT);

  //initialize all LED pins to output
  for(int i = 0; i < NUM_LED; i++){
    pinMode(ledPins[i], OUTPUT);
  }

  //get current sensitivity
  emgSaturationValue = sensitivities[lastSensitivitiesIndex];
}
```

```
//--------------------------------------------------------------------------------
//   Main loop
//
//   - Checks state of sesitivity button
//   - Checks state of default-gripper-state button
//   - Measure EMG
//   - Shows EMG strength on LED bar
//   - Sets angle of servo based on EMG strength and current mode (open/closed)
// ---------------------------------------------------------------------------------
void loop()
{

  //----------------------  Switch sensitivity ----------------------------------

  //check if button is pressed (HIGH)
  if (digitalRead(SENSITIVITY_BUTTON_PIN))
  {
    //turn off all the LEDs in LED bar
    for(int j = 0; j < NUM_LED; j++)
    {
      digitalWrite(ledPins[j], LOW);
    }

    //increment sensitivity index
    lastSensitivitiesIndex++;
    if(lastSensitivitiesIndex==NUM_LED)
    {
      lastSensitivitiesIndex = 0;
    }
```

```
  //get current sensitivity value
  emgSaturationValue = sensitivities[lastSensitivitiesIndex];

  //light up LED at lastSensitivitiesIndex position for visual feedback
  digitalWrite(ledPins[lastSensitivitiesIndex], HIGH);

  //wait user to release button
  while (digitalRead(SENSITIVITY_BUTTON_PIN))
  {
    delay(10);
  }
  //whait a bit more so that LED light feedback is always visible
  delay(100);
}


//----------------------------  Switch gripper default position open/close ---------

//check if enough time has passed for button contact to settle down
if((millis() - debouncerTimer) > 50)
{
  gripperStateButtonValue = digitalRead(GRIPPER_STATE_BUTTON_PIN);
  //if button is pressed
  if(gripperStateButtonValue == HIGH)
  {
    //if last time we checked button was not pressed
    if(userReleasedButton)
    {
      debouncerTimer = millis();
      //block button events untill user releases it
      userReleasedButton = 0;
```

```
      //toggle operation mode
      if(currentFunctionality == OPEN_MODE)
      {
        currentFunctionality = CLOSED_MODE;
      }
      else
      {
        currentFunctionality = OPEN_MODE;
      }
    }
   }
   else
   {
    userReleasedButton = 1;
   }
}



//------------------------------  Measure EMG ----------------------------------------

analogReadings = analogRead(A0);//read EMG value from analog input A0



//---------------------- Show EMG strength on LED ---------------------------------

//turn OFF all LEDs on LED bar
for(int j = 0; j < NUM_LED; j++)
{
  digitalWrite(ledPins[j], LOW);
}

//calculate what LEDs should be turned ON on the LED bar
```

```
analogReadings= constrain(analogReadings, 30, emgSaturationValue);
ledbarHeight = map(analogReadings, 30, emgSaturationValue, 0, NUM_LED);

//turn ON LEDs on the LED bar
for(int k = 0; k < ledbarHeight; k++)
{
  digitalWrite(ledPins[k], HIGH);
}



//---------- Drive Claw according to EMG strength ----------

//set new angle if enough time passed
if (millis() - oldTime > MINIMUM_SERVO_UPDATE_TIME)
{
  //calculate new angle for servo
  if(currentFunctionality == OPEN_MODE)
  {
    analogReadings = constrain(analogReadings, 40, emgSaturationValue);
    newDegree = map(analogReadings, 40 ,emgSaturationValue, 190, 105);
  }
  else
  {
    analogReadings = constrain(analogReadings, 120, emgSaturationValue);
    newDegree = map(analogReadings, 120 ,emgSaturationValue, 105, 190);
  }

  //check if we are in servo dead zone
  if(abs(newDegree-oldDegrees) > GRIPPER_MINIMUM_STEP)
  {
    //set new servo angle
    Gripper.write(newDegree);
```

```
    }
    oldTime = millis();
    oldDegrees = newDegree;
  }
}
```