

MOTOR SERVO CONTROL BY PYTHON USING A MOUSE X-AXIS POSITION

INTRODUCTION

The use of programming platforms such as Python and Arduino have become very popular in recent times, for this application we use both to be able to control through the computer mouse and why?, the most used form of input is the mouse that the keys in the way we interact with the computer, with this reason in this project has a simple and understandable way to control a servo motor with the movement of the mouse taking its x-axis position on the screen.

Objectives:

- Discover how setup a communication among the platforms Python 3 and Arduino.
- Get the position value of the mouse usb connected to the computer.
- Take control of one standard servo motor.

THEORETICAL FRAMEWORK

a) **Python.-** Python can be easy to pick up whether you're a first time programmer or you're experienced with other languages[1], for this particular project was used Python 3.6.2.



b) **Arduino.-** Arduino is an open-source electronics platform based on easy-to-use hardware and software. It's intended for anyone making interactive projects, [2], in the project was used Arduino 1.8.5 Version.



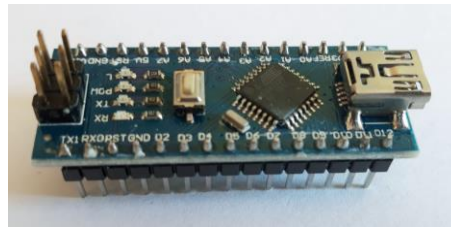
c) **USB.-** Stands for "Universal Serial Bus." USB is the most common type of computer port used in today's computers. It can be used to connect keyboards, mice(used for this application), game controllers, printers, scanners, digital cameras, and removable media drives, just to name a few,[3].



d) **Servo Motor.-** his high-torque standard servo can rotate approximately 180 degrees (90 in each direction). it can use any servo code, hardware or library to control these servos, [4], The servo used fo this project is SG-5010 Model.



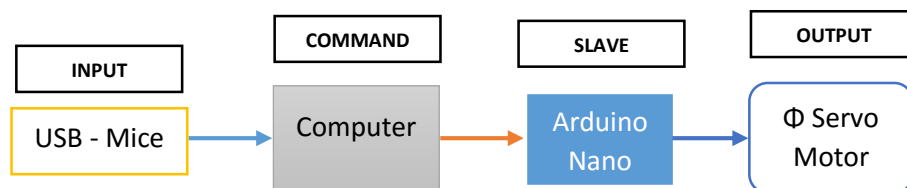
- e) **Arduino Nano.-** The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328P. It has more or less the same functionality of the Arduino Duemilanove, but in a different package. It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one,[6].



PROCEDURE

a) General Circuit and Flux Diagram

This a general view how the entire systems will work and as we can see has a input that is a usb-mice to get a positon values, and then in the computer using Python is processed correctly in orden that the values to sent to the Arduino Nano that has a code to receive those values and move the servo according the movement of the mice, shown in th Figure below.



b) Code in Arduino

In the Code we intruduce firts a Servo using a library *<Servo.h>*, then is choose a pin in the Arduino that is preset with pwm and this is *< servoPin >* then to setup the communication is usedthe command *< Serial.begin >* to start serial port for this project is used a 9600 baudios, later on is read the port that the values coming are Integer so for this section the command *<Serial.parseInt>* read the input sent by Python encoder and finally the servo motor keep your position, [7], more details in the Appendix (a) the IDE Arduino is shown below.

```
#include <Servo.h> //include the servo library

int pos = 0; //declare initial position of the servo
int servoPin = 9; //Decla Pin 9 for the servo
int servoDelay = 15; //delay to allow the servo to reach position;

Servo myServo; // called myServo object

void setup() {
  Serial.begin(9600); //start serial port
  myServo.attach(servoPin); //declare to which pin is the servo connected
}

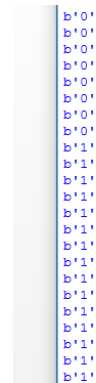
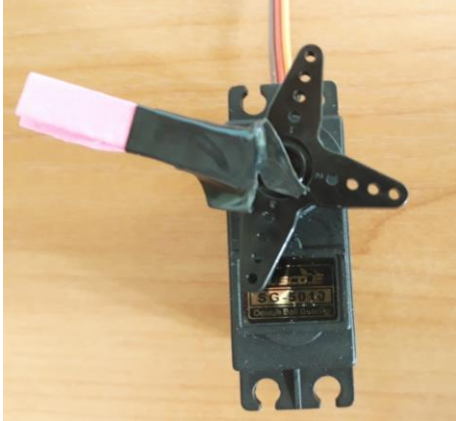
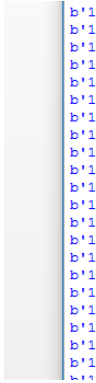
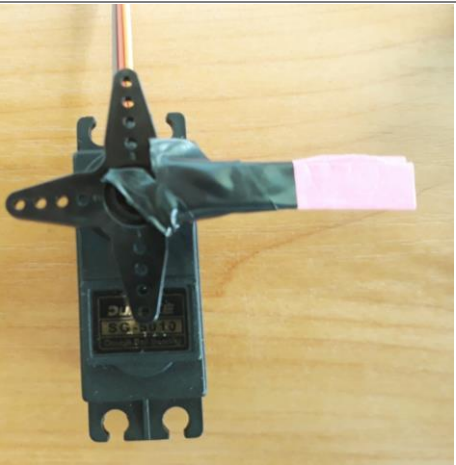
void loop() {
  while(Serial.available() > 0) {} //wait until information is received from the ser
  pos = Serial.parseInt(); //read the position from the servo
  if(pos > pos){
    myServo.write(pos); //write the position into the servo
    Serial.println(pos);
    delay(servoDelay); //give time to the servo to reach the position
  }
}
```

c) **Code in Python 3**

The libraries used are, “serial(to manage any usb port), time(time), pyautogui(to get postion values of the mice)”, and the commands used to enconder the values got from the mice, [8] are “srt.encoder(value)” and “object.write(value)” to be sure the value is printed byt the command “print(value)”, more details, Appendix (a).

```
Python3 - C:\Python3\Python3.py (1.8.2)
File Edit Shell Run Options Window Help
# For any USB systems
# Module to use USB ports
import serial
# Time
import time
# Module for programmatically controlling the m
import pyautogui
arduino = serial.Serial('COM6', 9600, timeout = 1)
time.sleep(2)
# Get the position of the mouse
# Get just a x and y and Map(0-1270 Screen Size to
# Conversion to Integer Values
# Connect is into string
# Encoding string
# Send data by the port selected
# Delay(1). If you have a servo with quick stop
# Get the actual position
while True:
    x, y = pyautogui.position()
    x = (x*0.161)
    x = (int(x))
    x = str(x)
    x = str.encode(x)
    arduino.write(x)
    time.sleep(1.5)
    print(x)
```

RESULTS

Mouse Pointer in the left side in the screen	Servo Motor 1 ⁰ Angle
	
Mouse Pointer in the right side in the screen	Servo Motor 177 ⁰ Angle
	

CONCLUTIONS

As a first conclusion we can say that the first objective was successfully achieved because we understood how the protocols and commands of each of the programs work with both Python and Arduino.

The position of the mouse usb, was difficult to know how works but in the end thank to the command “Endonce” in Python finally it was possible to send data to the Arduino.

There are few problems to control the servo motor because the arduino platform includes libraries that facilitate the use of the servo motor, noting that in order for the program to generate as in Python and Arduino the servo motor must be the most reactive motor with a cycle of smaller work and so be able to quickly see the reaction of the motor to the input of the mouse.

REFERENCES:

- [1] Python Organization, Queried in website: <https://www.python.org/>
- [2] Arduino Team, Queried in website: <https://www.arduino.cc/>
- [4] Dictionary, USB, Queried in website: <https://techterms.com/definition/usb>
- [5] Standard servo - TowerPro SG-5010 – 5010, Queried in website: <https://www.adafruit.com/product/155>
- [6] The Arduino, Queried in website: <https://store.arduino.cc/usa/arduino-nano>
- [7] Funtion Map, map(),Queried in website, <https://www.arduino.cc/reference/en/language/functions/math/map/>
- [8] Mouse Control Functions, The Screen and Mouse Position, Queried in website: <http://pyautogui.readthedocs.io/en/latest/mouse.html>

Appendices

a) Script in Arduino

Arduino Code - Servo Control	
#include <Servo.h>	// Include the servo library
int pos = 0;	// Declare initial position of the servo
int servoPin = 3;	// Declare pin for the servo
int servoDelay =15;	// Delay to allow the servo to reach position;
Servo myServo;	// Create a servo object called myServo
void setup() {	
Serial.begin(9600);	// Start serial port
myServo.attach(servoPin);	// Declare to which pin is the servo connected
}	
void loop() {	
while(Serial.available()>0){};	// Wait until information is received from the
	// Serial port
pos = Serial.parseInt();	// Read the position
if(pos=pos){	
myServo.write(pos);	// Write the position into the servo
Serial.println(pos);}	// Print position angle
delay(servoDelay);	// Give time to the servo to reach the position
}	

b) Script in Python 3

Python 3 Code - Mouse get position values	
#!/usr/bin/ env python	# For any OS systems
import serial	# Modele to use USB ports
import time	# Time
import pyautogui	# Module for programmatically #controlling the mouse and keyboard
arduino = serial.Serial('COM6', 9600, timeout = 1)	# Set up the Conection Arduino - PC
time.sleep(2)	# Wait for Arduino
while True:	
x, y = pyautogui.position();	# Get the position of the mouse
x = (x)*0.141;	# Get just x axis and map[0-1270 # Screen Size to 0 - 180 turning # angle]
x = (int(x))	# Covertion to Integer Values
x = str(x)	# Convert it into string
x = str.encode(x)	# Encoding strings
arduino.write(x)	# Send data by the port selected
time.sleep(1.5)	# Delay(ms), If you have a servo # with quick response decrease the # time
print(x)	# See the actual position