

# Práctica 5 - Control de servomotores

## Interfaces y periféricos para robots

Roberto Cadena Vega

### OBJETIVOS

Controlar la posición de un motor de corriente directa con un sensor de realimentación (servomotor).

### CONOCIMIENTOS PREVIOS

#### CONTROLADORES

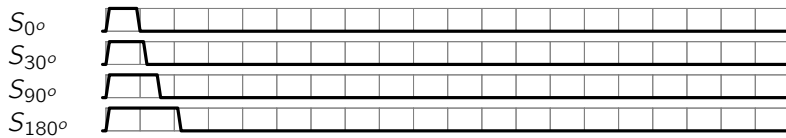
En esta práctica utilizaremos primero el servomotor de la manera en que el fabricante esperaba que lo usáramos, después lo modificaremos de tal manera en que podamos controlarlo completamente desde el Arduino.

En primer lugar debemos saber como funciona normalmente; para esto lo conectaremos de acuerdo a las instrucciones del fabricante, tomando cuidado de conectar el cable de señal al pin 13 de la tarjeta de desarrollo Arduino, para darle la señal de control por medio de este.

Ahora podemos subir el siguiente código a nuestro Arduino, con el que podremos modificar su posición por medio de comandos en el monitor serial del IDE de Arduino.

```
1 const int pin_servo = 13;
2 float pos = 90;
3 float t_on = 0;
4 float t_off = 0;
5
6 void setup() {
7   Serial.begin(9600);
8   pinMode(pin_servo, OUTPUT);
9 }
10 void loop() {
11   recibir_datos();
12   mandar_senal_servo();
13 }
14 void recibir_datos(){
15   if (Serial.available()) {
16     pos = Serial.readString().toFloat();
17     definir_senal_servo();
18   }
19 }
20
21
22
23 void definir_senal_servo(){
24   t_on = pos/150.0 + 0.9;
25   t_off = 3 - t_on;
26 }
27
28
29
30 void mandar_senal_servo(){
31   digitalWrite(pin_servo, HIGH);
32   delayMicroseconds(t_on*1000);
33   digitalWrite(pin_servo, LOW);
34   delayMicroseconds(t_off*1000);
35   delay(17);
36 }
```

En este código, la estructura general es que intentará recibir datos del puerto serial, y después mandará la señal al servomotor. Cuando recibe datos del puerto serial supondrá que es un número entre  $0^\circ$  y  $180^\circ$  y hará el cálculo de los retardos necesarios para dar la señal correcta al servomotor. Si analizamos este cálculo, podremos ver que cuando la posición es  $0^\circ$ , el tiempo que la señal permanecerá encendida es  $0.9ms$  y el tiempo que permanecerá apagada es  $2.1ms + 17ms = 19.1ms$ , es decir, una señal de  $20ms$  en total; cuando la posición es  $180^\circ$ , el tiempo de encendido es  $2.1ms$ , el tiempo de apagado es  $0.9ms + 17ms = 17.9ms$ , es decir, una señal de  $20ms$  en total.



Cabe añadir que si esta señal dura  $20ms$  en su totalidad, se tiene que enviar cada  $20ms$ , dependiendo de cada fabricante, es posible que el servomotor mantenga esta posición hasta que llegue una señal nueva, o bien, que si no llega una nueva señal después de  $20ms$ , el controlador del servomotor suponga que el motor debe de ser apagado y el servomotor no conservará su posición al aplicarse un par de torsión. También es posible que el servomotor utilice un esquema diferente de excitación con tiempos diferentes a los establecidos en este código, en este caso será necesario modificar estos tiempos para adecuarse a la información del datasheet del servomotor.

En este punto deberás llenar la primer tabla de la hoja de anotaciones, con los tiempos que le toma a tu servomotor moverse de la posición  $0^\circ$  a la posición  $180^\circ$  o de la posición  $180^\circ$  a la posición  $0^\circ$ , para obtener un tiempo promedio de la operación realizada por el servomotor.

### TRAYECTORIAS DEPENDIENTES DEL TIEMPO

Si ahora queremos medir el error que va a tener este servomotor en una cierta trayectoria, necesitamos cargar el siguiente código el cual dará una señal senoidal al servomotor con una frecuencia variable:

```

1  const int pin_servo = 13;
2  float pos = 0;
3  float t_on = 0;
4  float t_off = 0;
5  float frec = 1;
6  float omega = TWO_PI*frec;
7  float t = 0;
8
9  void setup() {
10   Serial.begin(9600);
11   pinMode(pin_servo, OUTPUT);
12 }
13 void loop() {
14   recibir_datos();
15   trayectoria();
16   definir_senal_servo();
17   mandar_senal_servo();
18 }
19 void recibir_datos(){
20   if (Serial.available()) {
21     frec = Serial.readString().toFloat();
22     Serial.println(frec);
23     definir_frecuencia();
24   }
25 }
26
27
28 void definir_senal_servo(){
29   t_on = pos/150 + 0.9;
30   t_off = 20 - t_on;
31 }
32
33
34 void mandar_senal_servo(){
35   digitalWrite(pin_servo, HIGH);
36   delayMicroseconds(t_on*1000);
37   digitalWrite(pin_servo, LOW);
38   delayMicroseconds(t_off*1000);
39   delay(17);
40 }
41
42
43 void definir_frecuencia(){
44   omega = TWO_PI*frec;
45 }
46
47
48 void trayectoria(){
49   t = millis()/1000.0;
50   pos = (sin(omega*t) + 1)*90;
51 }

```

Este código empezará dando una señal senoidal al servomotor, haciendo que se mueva de  $0^\circ$  hasta  $180^\circ$  y de regreso, con una frecuencia de  $1Hz$  para empezar, y con la posibilidad de enviar la frecuencia requerida por medio del monitor serial. Este código reutiliza elementos del pasado, pero antes de definir la posición a la que se debe mover el servomotor, primero calcula la posición por medio de una función senoidal que depende del tiempo que el Arduino ha permanecido encendido.

Si jugamos con este código, enviando valores por medio del puerto serial, podremos notar que mientras mayor sea la frecuencia de la señal senoidal, menor es el ángulo que el servomotor está recorriendo, no puede moverse lo suficientemente rápido antes de que tenga que regresar de nuevo.

En este punto deberás llenar la segunda tabla con el ángulo que recorre el servomotor dependiendo de la frecuencia dada.

### HACKING

**¡Peligro!** - A partir de este punto se harán modificaciones importantes al servomotor las cuales harán difícil el volver a resolver los puntos anteriores, por lo que es importante que hayas terminado las primeras dos tablas y hayas validado tu trabajo con el profesor, antes de empezar a trabajar en esta sección de la práctica.

Una vez que hemos caracterizado el desempeño de nuestro servomotor, podemos seguir adelante a el diseño de un controlador por medio de la tarjeta de desarrollo Arduino, para esto necesitaremos abrir la parte posterior del servomotor, retirar el controlador actual, desoldando o cortando los cables que entran a este y añadiendo longitud a los cables, los cuales deben ser, dos conductores del motor CD y tres conductores del potenciómetro; una vez que estos cables han sido alargados, se puede cerrar el servomotor de nuevo.

Los dos conductores del motor CD deben ser conectados al circuito de potencia adquirido (L293D o L298N), las señales de control para este circuito vendrán del Arduino y estarán marcadas como m1 y m2. Los tres conductores del potenciómetro serán conectados a 0V, 5V y al pin marcado como pot\_pin en el código; de tal manera que el Arduino detecte un voltaje en este pin entre 0V y 5V dependiendo de la posición del servomotor.

Una vez con nuestro circuito conectado, es momento de probar el primer código:

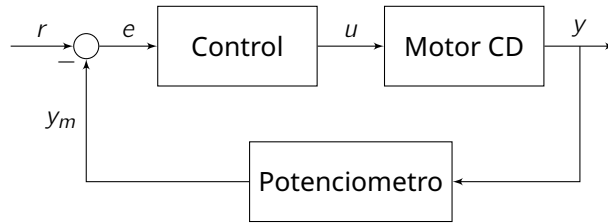
```

1 const int pot_pin = A0;
2 const int m1 = 10;
3 const int m2 = 11;
4
5 float kp = 1;
6 float sensor = 0;
7 float pos_des = 100;
8 float error = 0;
9 int senal = 0;
10
11 void setup(){
12   Serial.begin(9600);
13   pinMode(m1, OUTPUT);
14   pinMode(m2, OUTPUT);
15 }
16
17 void loop(){
18   recibir_datos();
19   sensor = analogRead(pot_pin);
20   Serial.println(sensor);
21   calcular_error();
22   calcular_senal();
23   if(senal < 0) reversa();
24   else avance();
25   delay(2);
26 }
27
28 void recibir_datos(){
29   if(Serial.available() > 0){
30     pos_des = Serial.readString().toFloat();
31   }
32 }
33
34 void calcular_error(){
35   error = pos_des - sensor;
36 }
37
38 void calcular_senal(){
39   senal = int(max(min(kp*error, 255), -255));
40 }
41
42 void reversa(){
43   analogWrite(m1, 0);
44   analogWrite(m2, abs(senal));
45 }
46
47 void avance(){
48   analogWrite(m2, 0);
49   analogWrite(m1, senal);
50 }

```

Este código intentará recibir la referencia de la posición deseada del puerto serial, leerá la posición del potenciómetro, mandará este dato por el puerto serial, calculará

el error entre la posición deseada y la posición real, calculará la señal necesaria para mover el motor a la posición deseada, y por ultimo, ira hacia adelante o hacia atras, dependiendo de si la señal necesaria es positiva o negativa. Si dibujamos un diagrama de bloques de la secuencia de información se verá asi:



En el código, el control esta determinado por  $k_p$ , de tal manera que esta cantidad determinará el comportamiento del sistema.

Si jugamos con el sistema asi, podemos enviar la posicion de refencia por medio del monitor serial, tambien podemos modificar la ganancia  $k_p$ , de tal manera en que el sistema sea mas rapido o lento, estable o criticamente estable, subamortiguado o sobreamortiguado.

En este punto deberás llenar la tercer tabla, sintonizando el controlador  $k_p$  para obtener el menor tiempo posible en mover el motor de la posición  $0^\circ$  a  $180^\circ$  o bisceversa.

### CONTROL PD

Hasta el momento hemos utilizado un control proporcional simple, pero podemos implementar un control un poco mas complejo con pocos calculos; el control PD tan solo requiere que calculemos el error y la velocidad del error que es equivalente a la velocidad del motor.

```

1  const int pot_pin = A0;
2  const int m1 = 10;
3  const int m2 = 11;
4
5  float kp = 10;
6  float kd = 2.5;
7  float sensor = 0;
8  float pos_des = 100;
9  float error = 0;
10 float error_anterior = 0;
11 int senal = 0;
12
13 void setup(){
14   Serial.begin(9600);
15   pinMode(m1, OUTPUT);
16   pinMode(m2, OUTPUT);
17 }
18
19 void loop(){
20   recibir_datos();
21   sensor = analogRead(pot_pin);
22   Serial.println(sensor);
23   calcular_error();
24   calcular_senal();
25   if(senal < 0) reversa();
26   else avance();
27   delay(2);
28 }
29
30 void recibir_datos(){
31   if(Serial.available() > 0){
32     pos_des = Serial.readString().toFloat();
33   }
34 }
35
36 void calcular_error(){
37   error_anterior = error;
38   error = pos_des - sensor;
39 }
40
41 void calcular_senal(){
42   float u = kp*error + kd*(error - error_anterior);
43   senal = int(max(min(u, 255), -255));
44 }
45
46 void reversa(){
47   analogWrite(m1, 0);
48   analogWrite(m2, abs(senal));
49 }
50
51 void avance(){
52   analogWrite(m2, 0);
53   analogWrite(m1, senal);
54 }
  
```

En este punto deberás llenar la tabla cuatro, la cual te ayudará a sintonizar el controlador PD utilizando el mismo procedimiento de la anterior.

### EQUIPO

El siguiente equipo será proporcionado por el laboratorio, siempre y cuando lleguen en los primeros 15 minutos de la práctica, y hagan el vale conteniendo el siguiente equipo (exceptuando las pinzas).

- Fuente de alimentación
- Osciloscopio
- Cables BNC - caimán
- Cables banana - caimán
- Cables de alimentación
- Multímetro
- Pinzas
- Transportador

### MATERIALES

- Protoboard
- Cables
- Motor CD
- LED
- L293D, L298N, etc.

### DESARROLLO

Diseña los circuitos requeridos y toma las mediciones necesarias para llenar la hoja de anotaciones y realiza el análisis cuantitativo del error de cada controlador.

### CONCLUSIONES

El alumno deberá describir sus conclusiones al final de su reporte de práctica.

## HOJA DE ANOTACIONES

1. Llena la tabla de abajo, utilizando el primer código para validar el tiempo necesario para que tu servomotor se mueva de la posición  $0^\circ$  a la posición  $180^\circ$  o de la posición  $180^\circ$  a la posición  $0^\circ$ .

Intento	1	2	3	4	5	6	7	8	9	10
$t$										

2. Llena la tabla de abajo, utilizando el segundo código para validar el angulo recorrido por el servomotor en cada una de las frecuencias descritas abajo.

f (Hz)	0.25	0.5	0.75	1.00	1.25	1.5	1.75	2.0	4.00	8.00
$\Delta\theta$										

3. Diseña el circuito necesario para conectar tu servomotor al Arduino por medio del circuito de potencia que hayas adquirido.

4. Llena la tabla de abajo, utilizando el tercer código para sintonizar el controlador, minimizando el tiempo necesario para que tu servomotor se mueva de la posición  $0^\circ$  a la posición  $180^\circ$  o de la posición  $180^\circ$  a la posición  $0^\circ$ .

Intento	1	2	3	4	5
kp					
$t_1$					
$t_2$					
$t_3$					
$t_4$					

5. Llena la tabla de abajo, utilizando el tercer código para sintonizar el controlador, minimizando el tiempo necesario para que tu servomotor se mueva de la posición  $0^\circ$  a la posición  $180^\circ$  o de la posición  $180^\circ$  a la posición  $0^\circ$ .

Intento	1	2	3	4	5
kp					
k $\dot{d}$					
$t_1$					
$t_2$					
$t_3$					
$t_4$					

Integrantes del equipo:

Revisó:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_