https://repository.uaeh.edu.mx/revistas/index.php/icbi/issue/archive
Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI

Pädi

ISSN: 2007-6363

Publicación Semestral Pädi Vol. 8 No. 15 (2020) 14-21

Diseño e Implementación de una Plataforma de Control Simplificada para la Retroalimentación de un Plotter CNC

Design and Implementation of a Simplified Control Platform for the Feedback of a CNC Plotter

M. Morales-Morfin[a,*], G. Loreto-Gómez[b]

[a] *Posgrado CIATEQ, A.C., Av. del Retablo, No 150, Col. Constituyentes Fovissste, C.P. 76150, Querétaro, Qro., México.*
[b] *Departamento de Ingeniería Mecatrónica, Instituto Tecnológico Superior de Uruapan, Carretera Uruapan-Carapan, No 5555, Col. La Basilia, C.P 60015, Uruapan, Mich., México.*

Resumen

En este artículo se realiza el diseño e implementación de una plataforma de control simplificada para el control retroalimentado de un plotter CNC; se programa un controlador PID para el control de los servomotores el cual está basado en la programación a bloques en Simulink-Matlab™. La retroalimentación se hace mediante la tarjeta de desarrollo del microcontrolador Atmel 2560 programado para adquirir los datos del encoder de un servomotor UNITEC. La validación experimental muestra que la integración de los diferentes módulos de muestreo en un bloque programado en Simulink-Matlab™, simplifica el diseño electrónico de la conversión de señales, implementando por software muchas rutinas de conversión de señales, que antes no era posible de manera directa, facilitando a los investigadores el desarrollo y prueba de nuevos algoritmos de control en la plataforma de manera directa y en tiempo real.

Palabras Clave:
Control PID, Servomotor, Atmel 2560, Simulink-Matlab™, CNC retroalimentado.

Abstract

In this paper the design and implementation of a simplified control platform for the feedback control of a CNC plotter is carried out; A PID controller is programmed for servo motor control which is based on block programming in Simulink-Matlab™. The feedback is done through the Atmel 2560 microcontroller development card programmed to acquire the encoder data of a UNITEC servomotor. Experimental validation shows that the integration of the different sampling modules in a block programmed in Simulink-Matlab™, simplifies the electronic design of signal conversion, by software implementing many signal conversion routines, which previously was not possible directly, making it easier for researchers to develop and test new control algorithms on the platform directly and in real time.

Keywords:
PID Control, Servomotor, Atmel 2560, Simulink-Matlab™, CNC feedback.

1. Introducción

Los motores eléctricos son fundamentales en las aplicaciones en las que se requiere llevarse a cabo un movimiento mecánico. Estos se dividen principalmente en dos tipos: los motores de corriente continua (DC) y de corriente alterna (AC), según el tipo de corriente que utilicen para funcionar. Los motores de DC de imanes permanentes tienen sus polos de campo alimentados por imanes que son del tipo semicírculos y están ensamblados sobre la carcasa del estator para proporcionar uno o más pares de polos. Sin estructura de devanado de campo los motores de DC tienden a ser de tamaño más compacto y tener mayor eficiencia, además de tener una operación más fría con una construcción totalmente cerrada. Si los motores de imanes permanentes no se sobrecargan, no ocurre la desmagnetización del flujo en el entrehierro y la característica par-velocidad se queda como una curva lineal, y

*Autor para la correspondencia: marcelmmorfin@gmail.com
Correo electrónico: marcelmmorfin@gmail.com (M. Morales-Morfin), dr.gloreto@hotmail.com (G. Loreto-Gómez)

Fecha de recepción: 30/09/2019 Fecha de aceptación: 15/11/2019 Fecha de publicación: 05/07/2020
DOI: https://doi.org/10.29057/icbi.v8i15.5003

el control del par-velocidad se logra por medio del ajuste de voltaje de armadura. (Enríquez Harper, (2005)).

Actualmente, los CNC´s comerciales de tres ejes utilizan en su diseño motores paso a paso del estándar NEMA 17, con todas las ventajas y desventajas que tienen como son: su baja eficiencia, lentitud, poca fuerza a altas velocidades y la ausencia de retroalimentación, los motores paso a paso no conocen su posición en todo momento, por lo que no se puede tener una seguridad al 100% de su operación y no pueden corregirse a menos que cuenten con un sistema desarrollado a medida para hacer estas correcciones. Los servomotores son en general un conjunto de cuatro cosas: un motor de corriente continua, un conjunto de engranajes, un circuito de control y un sensor de posición. La posición de los servomotores se puede controlar con mayor precisión que los de motores de DC, y por lo general tienen tres cables: alimentación, tierra y control. La alimentación para servomotores se aplica constantemente y los grados de giro se controlan con un circuito de control de servo regulación. Los servomotores están diseñados para tareas más específicas en las que la posición debe definirse con precisión como el control del timón en un barco o mover un brazo robótico dentro de un cierto rango (Staticbboards, 2019).

El tiempo de posicionamiento se reduce gracias a la reducción de la inercia del rotor lo que permite alcanzar altas velocidades en tiempos reducidos y, por otra parte, la posibilidad de hacer girar un motor con una velocidad nominal de 2000 rpm a una velocidad de rotación máxima de 4500 rpm. Las características principales de este tipo de motores son:

- Prestaciones y par elevado
- Fiabilidad de funcionamiento
- Bajo mantenimiento
- Gran exactitud en el control de velocidad y posición
- Capacidad de velocidades muy altas
- Pérdidas en el rotor muy bajas
- Rotor con poca inercia
- Construcción cerrada, útil para trabajar en ambientes sucios
- Amplia gama de potencias (de 100 w a 300 Kw)

El fabricante UNITEC Corporation Japan proporciona datos básicos del servomotor DTME-2730A que se utiliza en el prototipo, los cuales se muestran en la Tabla 1.

Tabla 1: Características Técnicas del servomotor DTME-2730A

UNITEC servo DC motor con Encoder	
Tipo	DTME-2730A
N	15
Volts	24V
AMPS	2.1
Watts	30
R.P.M.	800
Diámetro del eje	10mm

Los servomotores de estructura compacta incorporan dentro de la misma un encoder absoluto o incremental, el cual suministra información del estado del proceso al controlador (Jesús F. Mora, (2003)). Esta es la razón principal para usarlos en este trabajo. En la Figura 1, se muestra la estructura interna del encoder del servomotor.

Figura 1: Servomotor con encoder incremental.

El presente trabajo se organiza de manera siguiente: en la sección 2 se detalla la teoría preliminar y las tarjetas que se utilizan en el desarrollo del trabajo. En la Sección 3 se describen los resultados principales obtenidos, a partir del diseño y la construcción de la plataforma, además se explica la adquisición de datos mediante el programa Simulink-Matlab™. Finalmente, el trabajo termina con las conclusiones en la Sección 4.

2. Preliminares

Dado que este trabajo se basa en el diseño de un plotter CNC de tres ejes con servomotores y controlado a través de una interfaz en Matlab, es indispensable presentar los conceptos que se van a manejar en el artículo.

2.1 Control PID

Como casi todos los controladores PID se ajustan en el sitio, en la literatura se han propuesto muchos tipos diferentes de reglas de sintonización, que permiten llevar a cabo una sintonización delicada y fina de los controladores PID en el sitio. Asimismo, se han desarrollado métodos automáticos de sintonización y algunos de los controladores PID poseen capacidad de sintonización automática en línea. Actualmente se usan en la industria formas modificadas del control PID, tales como el control I-PD y el control PID con dos grados de libertad (Ogata, (2010)).

Controlador PID calcula el error $e(t)$ existente entre el valor deseado $e_d(t)$ y la salida $y(t)$ de un sistema, es decir, $e(t) = e_d(t) - \mathrm{y}(t)$; con lo cual se genera una señal de control formada por las siguientes tres acciones (Bolton, (2001)):

Acción proporcional (P): Da una salida del controlador que es proporcional al error, la cual puede ser descrita en el dominio de la frecuencia tal que:

$$C_p(s) = \frac{U_1(S)}{E(s)} = K_p, \qquad (1)$$

Donde K_p es la ganancia proporcional ajustable (Pérez M. A., Pérez and Pérez, (2008)).

Acción integral (I): Da una salida del controlador proporcional al error acumulado, lo que implica una respuesta de control lenta. Esta acción se define en el dominio de la

frecuencia con la siguiente función de transferencia (Pérez M. A., Pérez and Pérez, (2008)).

$$C_i(s) = \frac{U_2(S)}{E(s)} = \frac{K_i}{s}, \quad (2)$$

donde K_i es la ganancia integral ajustable.

Acción derivativa (D): Actúa cuando hay un cambio en el valor absoluto del error, corrigiendo sólo los errores en la etapa transitoria. Al ser una acción predictible su respuesta es rápida Esta acción se define en el dominio de la frecuencia como

$$C_d(s) = \frac{U_3(S)}{E(s)} = K_d s, \quad (3)$$

Donde K_d es la ganancia derivativa ajustable (Pérez M. A., Pérez and Pérez, (2008)).

Aquí, la acción proporcional, tal como su nombre lo indica, depende proporcionalmente del error; la parte integral provee robustez ante perturbaciones constantes, eliminando el error en estado estacionario; mientras que la acción derivativa garantiza amortiguamiento para el sistema en lazo cerrado como se muestra en la Figura 2.

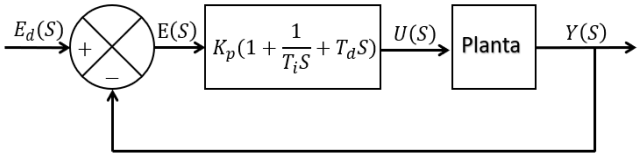

Figura 2: Control PID de una planta.

Donde $E_d(S)$ es el valor deseado, $E(S)$ es el error, $U(s) = U_1(S) + U_2(S) + U_3(S)$ es la salida del controlador y $Y(s)$ la salida del sistema o el valor real, (Bolton, (2001)).

La utilidad de los controles PID estriba en que se aplican en forma casi general a la mayoría de los sistemas de control. En el campo de los sistemas para control de procesos, es un hecho bien conocido que los esquemas de control PID básicos y modificados han demostrado su utilidad para aportar un control satisfactorio, aunque tal vez en muchas situaciones específicas no aporten un control óptimo (Åström y Hägglund, (1995)).

2.2 Método de Ziegler-Nichols

A continuación, se describe la sintonización para la ley de Control en el dominio del tiempo:

$$u(t) = K_p\left(e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right)$$

mediante el segundo método propuesto por Ziegler-Nichols.

1. En primer lugar, se establecen los parámetros Ti = 1 y Td = 0.
2. Aplicando únicamente la ganancia Kp, se incrementa este valor desde cero hasta un valor crítico Kcr, es decir, cuando la salida del sistema presenta oscilaciones.
3. Posteriormente se determina el periodo crítico Pcr correspondiente a las oscilaciones obtenidas en el paso anterior, a partir de la cual se calculan las ganancias del controlador PID, de la misma manera es posible calcular las ganancias para el control proporcional (P) y el control proporcional integral (PI), tal como se muestra en la Tabla 2 (Ziegler and Nichols, 1942).

Tabla 2: Sintonización de controladores P, PI y PID por el segundo método de Ziegler-Nichols

Tipo de controlador	K_p	T_i	T_d
P	$0.5K_{cr}$		
PI	$0.45\ K_{cr}$	$0.83P_{cr}$	
PID	$0.6K_{cr}$	$0.5P_{cr}$	$0.125P_{cr}$

2.3 Servomotores

En 1934, Hazen, introdujo el término servomecanismos para los sistemas de control de posición, analizó el diseño de los servomecanismos con relés, capaces de seguir con precisión una entrada cambiante. (Ogata, 2010).

Los servomotores son motores DC que incorporan un sensor de posición con un grado de resolución y una velocidad de rotación del eje de 2000 rpm. El modelo de los servomotores como actuadores del sistema, se define teniendo en cuenta los parámetros eléctricos de un motor DC y la caja reductora de velocidad que lo compone. Un esquema representativo se muestra en la Figura 3 (Hurbain, 2007) y (Valluru y Singh, (2016).

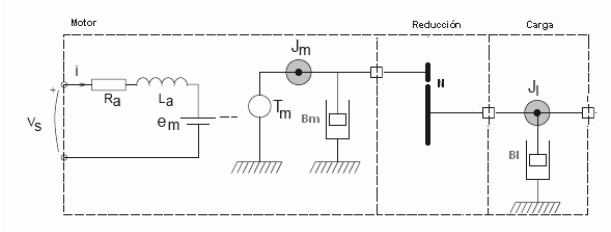

Figura 3: Diagrama esquemático de un Servomotor.

Los parámetros de la Figura 3 corresponden a:

R_a: Resistencia de armadura [Ω]

L_a: La: Inductancia de armadura [H]

K_e: Constante de fuerza contra electromotriz $[V/rad/s]$

K_T: Constante de torque $[Nm/A]$

J_m: Momento de Inercia del Motor $[Kgm^2]$

B_m: Coeficiente de fricción viscosa del motor $[Nm/rad/s]$

N: Relación de reducción del motor

J_l: Momento de Inercia de la carga $[Kgm^2]$.

B_l: Coeficiente de fricción viscosa de la carga motor $[Nm/rad/s]$

2.4 Tarjeta de desarrollo Arduino Mega 2560

El Arduino Mega 2560 es una placa de desarrollo basada en el microcontrolador ATmega2560. Tiene 54 entradas/salidas digitales (de las cuales 15 pueden ser usadas como salidas PWM), 16 entradas analógicas, 4 UARTs, un cristal de 16Mhz, conexión USB, jack para alimentación DC, conector ICSP, y un botón de reseteo. La placa Mega 2560 es compatible con la mayoría de shields compatibles para Arduino UNO. (Arduino Home, 2019).

2.5 Driver para servomotor VNH5019

Este driver de servomotor funciona de 5,5 a 24V y puede suministrar 12A continuos con picos de 30A. Funciona con niveles lógicos de 2.5 a 5 V, admite PWM ultrasónico (hasta 20 kHz) y presenta retroalimentación de detección de corriente (un voltaje analógico proporcional a la corriente del motor). Junto con la protección incorporada contra voltaje inverso, sobrevoltaje, bajo voltaje, sobretemperatura y sobrecorriente, estas características hacen de este producto un excelente controlador de servomotores de uso general. (Pololu Robotics & Electronics, 2019).

3. Resultados principales

En la presente sección se describe el diseño y la construcción del prototipo para el control del Servomotor UNITEC considerando las conexiones electrónicas con el driver VNH5019, mediante la tarjeta Arduino Mega 2560, junto con el programa Simulink-Matlab™.

3.1. Diseño y construcción de la plataforma

En la Figura 4 se muestra el circuito de conexiones implementado para el control del servomotor.

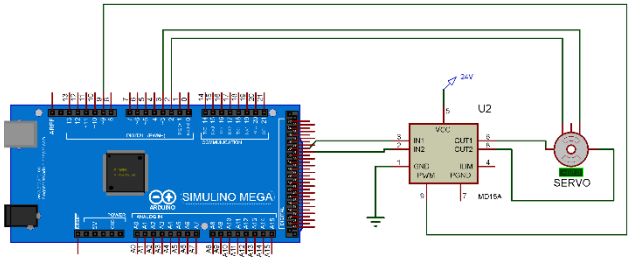

Figura 4: Diagrama de conexiones.

En el diseño del diagrama se puede observar la optimización final a la que se llegó, esto debido a que en las últimas actualizaciones de las herramientas de desarrollo en Matlab con arduino, se pueden implementar por software muchas rutinas de conversión de señales, que antes no era posible implementar de manera directa. Quedando en manos de los investigadores el desarrollo y prueba de los algoritmos de control en la plataformas.

3.2. Diseño del controlador PID para su implementación en Simulink-Matlab™.

Para empezar el diseño de los bloques de Simulink, se definen las condiciones iniciales para el sentido de giro y la potencia aplicada al motor mediante la señal PWM, una vez que el sistema está corriendo, se cierra el lazo de control y se introduce la posición deseada para obtener la señal de error mediante la señal de salida retroalimentada. Dicha señal, se acondiciona mediante el controlador PID sintonizado previamente y el resultado sirve como parámetro de referencia a la entrada PWM del motor y cuya salida del encoder proporciona la posición angular real, como se muestra en la Figura 5.

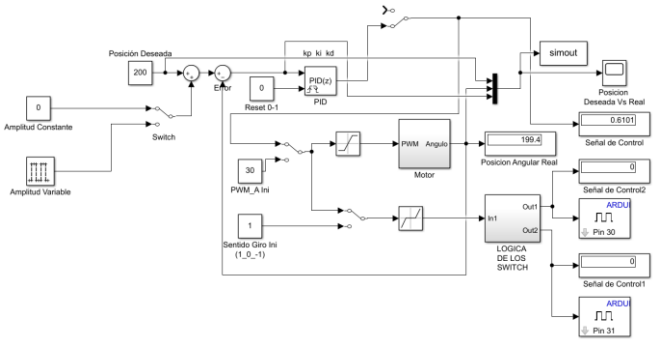

Figura 5: Diagrama a bloques para el controlador PID en Simulink.

Como puede observarse en el diagrama a bloques, la configuración es muy sencilla pero funcional.

Cuando se compila el diagrama de bloques en Simulink-Matlab™ se conecta al arduino mediante el puerto USB, para definir las condiciones iniciales tanto de giro y potencia aplicada al motor mediante el driver MD15A en el sistema real, leer la señal del encoder del servomotor para conocer su posición actual y enviarlas al controlador PID para cerrar el lazo de control. Una vez que el sistema está estable, es posible cambiar los parámetros para lograr la posición deseada en el eje que se esté trabajando. El código generado para el Arduino Mega, se muestra en el Apéndice A.

3.3 Análisis de la señal de Salida

Debido a la presintonización del controlador PID aplicando la metodología propuesta por el segundo método de Ziegler-Nichols, se consideraron los siguientes valores de las ganancias:

K_p	0.01
K_i	0.005
K_d	0

Estos valores se implementaron en el controlador PID en la herramienta Simulink-Matlab™ y junto con la tarjeta Arduino mega 2560, se obtuvo la respuesta del sistema en lazo cerrado, la cual se muestra en la Figura 6.

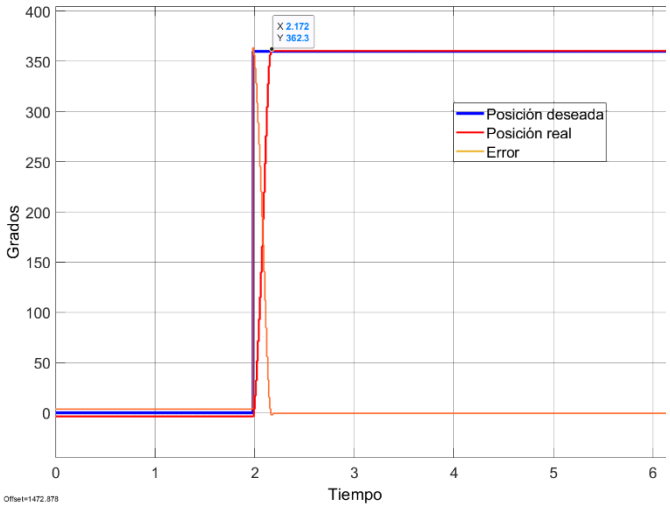

Figura 6: Salida del Sistema en lazo cerrado con el controlador PID.

Como se observa en la Figura 6, el experimento consistió en cambiar el valor del ángulo de posición inicial del motor de 0^o hasta un valor final de 360^o, obteniéndose una respuesta muy rápida en el seguimiento de la señal de referencia, lográndose en un tiempo de 0.172 segundos. El error relativo se calcula mediante el cociente entre el error absoluto de la posición deseada y la posición real que es de 2.3 grados y la posición real que es de 360 grados, obteniéndose un error relativo porcentual del 0.63%. Otro experimento consistió en introducir una perturbación al sistema que está en estado estable.

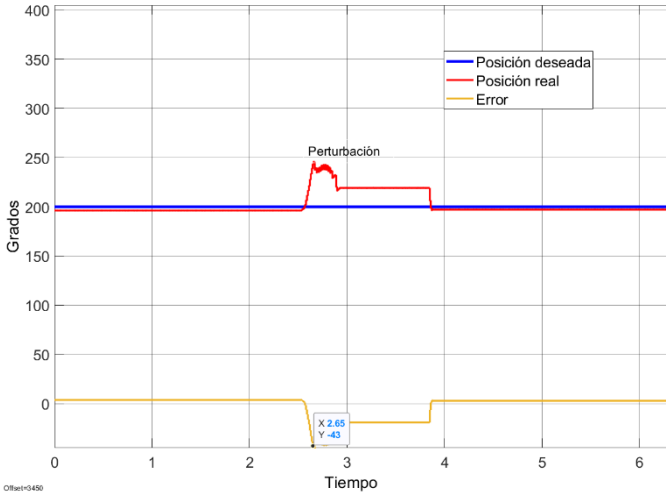

Figura 7: Introducción de una perturbación al sistema.

En la Figura 7, se observa que el controlador tuvo una perturbación de 43^o, saliendo de su posición estable, y debido a esto, el controlador hizo la corrección en dos etapas, lográndose el estado estable.

3.5. Implementación de la plataforma de control en un plotter CNC

Una vez que se prueba el controlador PID en un solo servomotor, este se replica dos veces para obtener la plataforma de control completa para el CNC, pudiéndose controlar los ejes X e Y en forma simultánea. También fue necesario desarrollar un generador de trayectorias y llevar a cabo la cinemática del plotter CNC, como se muestra en la Figura 8.

La primera etapa para el seguimiento de trayectoria en los ejes X, Y y la prueba de los controladores, corresponde a un planificador de trayectoria; con este, se obtienen las coordenadas x, y provenientes de la ecuación paramétrica de una circunferencia:

$$p = (x, y) = (x_c + r * \sin\theta, y_c + r * \cos\theta)$$

Siendo x_c, y_c el centro y θ el ángulo del punto expresado en radianes.

La cinemática inversa busca encontrar los valores (q_1, q_2), que tienen que tomar las articulaciones de un robot para que su elemento final se encuentre en una posición y orientación dada.

Debido a que la configuración del plotter CNC es cartesiana, está, no presenta alto grado de complejidad para determinar la cinemática inversa, simplemente es necesario saber cuáles son las coordenadas en el plano cartesiano en las cuales se encuentra ubicado el objeto y con estos valores se determina que distancia se debe desplazar cada uno de los eslabones. Esto es,

$$q_1 = \frac{x}{k}$$
$$q_2 = \frac{y}{k}$$

Donde k es la relación entre el desplazamiento angular y lineal.

La cinemática directa se encarga de determinar la posición del elemento final con respecto al sistema de referencia, conocidos los valores de las articulaciones y los parámetros geométricos del robot. De la misma forma, dado que la configuración del robot es cartesiana,

$$x = kq_1$$
$$y = kq_2$$

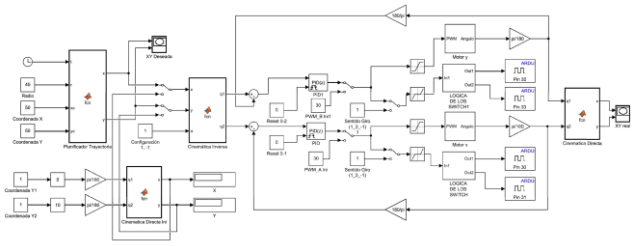

Figura 8: Diagrama a bloques para el seguimiento de una trayectoria en los ejes XY.

Para probar el seguimiento de una trayectoria en la plataforma, se genera un círculo introduciendo como condiciones iniciales el valor del radio de 45 y los valores (50,50) del centro y se hace un cambio brusco a un radio menor de 35, manteniendo el centro constante, resultando la trayectoria que se muestra en la Figura 9.

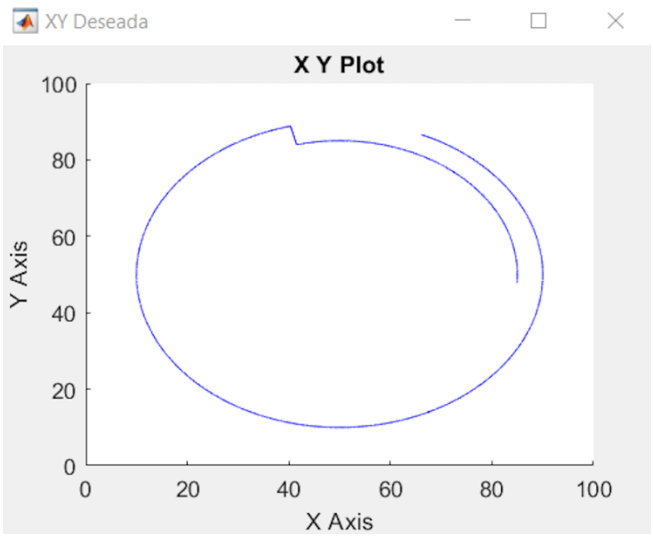

Figura 9: Trayectoria XY deseada.

Una vez que se activa el controlador para que siga la trayectoria, en la Figura 10, se puede visualizar que cuando se realiza un cambio brusco, el ajuste de la trayectoria es más lenta, este comportamiento ya se había observado cuando se introdujo una perturbación en el controlador, pero finalmente se logra el seguimiento adecuado de la trayectoria, como se muestra en la Figura 10.

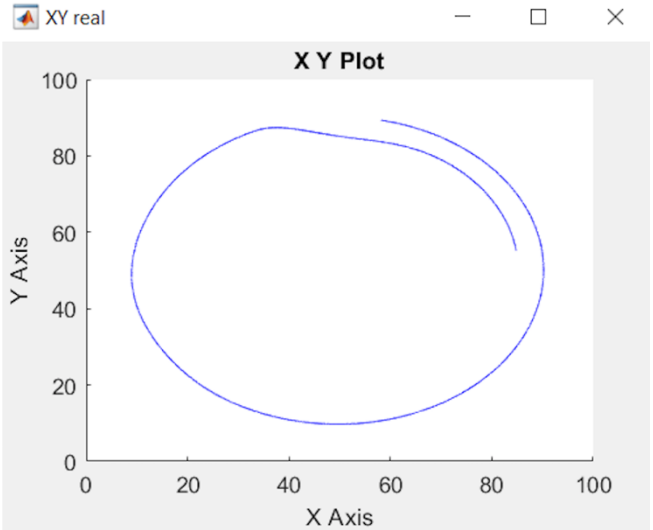

Figura 10: Trayectoria XY real.

En la Figura 11, se muestra la implementación final de la plataforma y el resultado obtenido del fresado del seguimiento de trayectoria para cambio del radio de una circunferencia.

Una de las ventajas del sistema propuesto es que permite ir validando de forma iterativa la aplicación de métodos de sintonización más robustos que vayan dando mejores desempeños, fortaleza de la plataforma en la que pueden probarse rápidamente, sin la necesidad de un proceso de programación. De esta manera, se logran obtener mejores resultados, Figura 12, bajo la misma condición de prueba anterior.

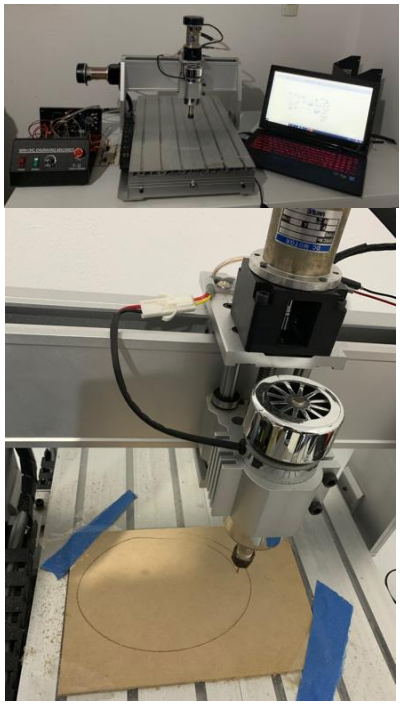

Figura 11: Plotter CNC con el control de servomotor implementado.

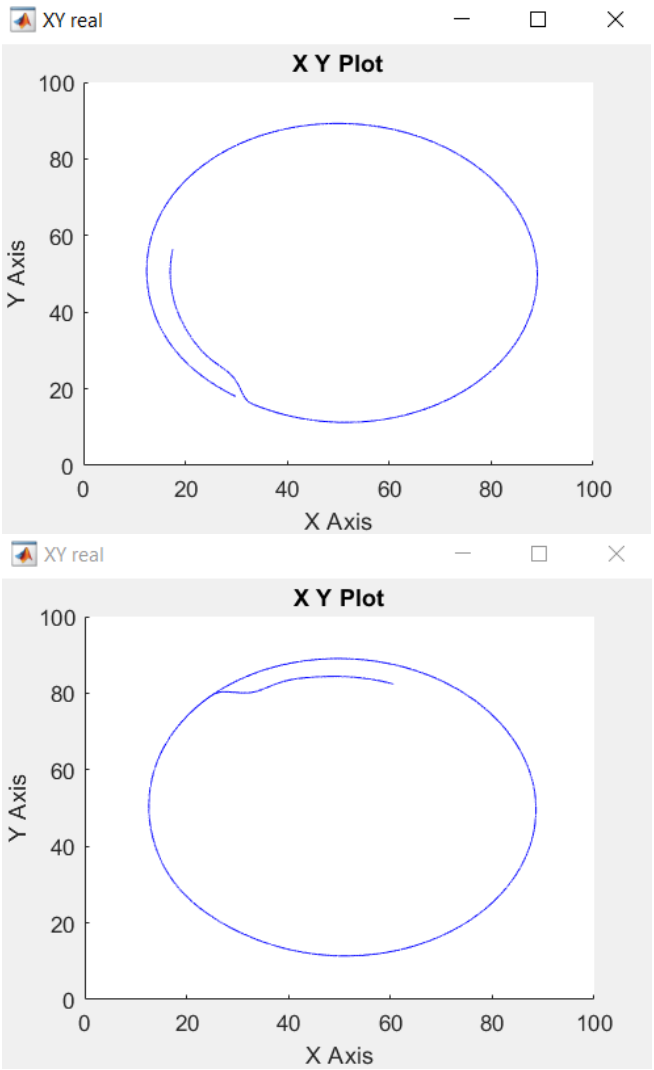

Figura 12: Otras trayectorias XY reales mejorando el tiempo de respuesta.

4. Conclusiones

Se ha diseñado y construido una plataforma de control con pocos componentes aplicado a un plotter CNC que puede ser explotado probando diferentes algoritmos de control en el entorno Simulink-Matlab™ en tiempo real.

Mediante la implementación de la ley de control PID en la plataforma desarrollada se han obtenido resultados experimentales aceptables para el control de posición de cada uno de los ejes del plotter CNC. Los resultados obtenidos son factibles para la implementación en tiempo real, ya que el algoritmo no utiliza operaciones matriciales que generan alto costo computacional.

Aplicando la metodología propuesta es relativamente sencillo sintonizar un controlador PID para el servomotor, debido a que se redujo considerablemente los tiempos de diseño electrónico, al incluir módulos de Simulink-Matlab™ amigables con el usuario.

Diferentes investigadores han implementado algoritmos de control para servomotores, ejemplo de ellos son: algoritmo de control por modo deslizante (Chang-Mumañ, Francisco; Mazaira-Morales, Israel, (2010)) y algoritmos de control inteligente en la plataforma de Simulink-Matlab™ y embebido en la tarjeta STM32F4 Discovery (Andino Alberca, Cristian Adrián, Rodríguez Sánchez, Diego Xavier (2016)). También en los últimos años, se han desarrollado controladores con mejores prestaciones haciendo uso de la tecnología FPGA, incluyendo módulos independientes para la conmutación electrónica de cada motor, así como módulos para el control de su posición (Martínez Fernández, G. A., Cruz Contreras, A., Hernández Herrera, V. G., & Márquez Olivera, M. V. (2017)). Sin embargo, la implementación de estos algoritmos genera sistemas complejos, difíciles de mantener y actualizar. Con la implementación que se realiza en este artículo, es posible facilitar a los investigadores el desarrollo y prueba de nuevos algoritmos de control para servomecanismos, de manera directa y en tiempo real, ya que solo basta cambiar el bloque de Simulink™ en donde se encuentra implementado el controlador y con la ventaja de que en la misma plataforma se acondicionan las señales que se requieren para su correcto funcionamiento.

Agradecimientos

Este trabajo ha sido realizado parcialmente gracias al apoyo del Consejo Nacional de Ciencia y Tecnología (Conacyt) y el Instituto Tecnológico Superior de Uruapan.

Referencias

Andino Alberca, Cristian Adrián, Rodríguez Sánchez, Diego Xavier (2016). Diseño e implementación de algoritmos de control inteligente para un robot Phoenix tipo hexápodo mediante la tarjeta STM32F4 discovery y simulink de matlab. Universidad de las Fuerzas Armadas ESPE. Matriz Sangolquí.

Arduino Home. (17 de Septiembre de 2019). Obtenido de Arduino.cc: https://www.arduino.cc

Åström, J., Hägglund, T., 1995. Pid controllers: theory, design, and tuning. Isa Research Triangle Park.

Bolton, W., 2001. Ingeniería de control, segunda Edición.

Cordero, E., 2016. Diseño y construcción de un prototipo de sistema motor-hélice-balancín.

Chang-Mumañ, Francisco; Mazaira-Morales, Israel, 2010. Control de un servomotor de cd. un enfoque algebraico diferencial. Ciencia en su PC, Núm. 3, julio-septiembre, 2010, pp. 1-10.

Enríquez Harper, 2005. Maquinas Electricas, primera edición. Editorial Limusa.

Hurbain P., Hurbain P. y Gasperi M. (2007). NXT Motor. The international Fan-Created Lego® Users Group Network. Extraído del World Wide Web:http://web.mac.com/ryo_watanabe/iWeb/Ryo%27s%20Holiday/NXT%20Motor.html.

Jesús F. Mora, 2003. Servoaccionamientos, segunda Edición. Editorial Mc Graw Hill.

Martínez Fernández, G. A., Cruz Contreras, A., Hernández Herrera, V. G., & Márquez Olivera, M. V. (2017). Controlador multieje de posicionamiento de servomotores BLDC implementado en FPGA. Pistas Educativas, Vol. 39 Num. 125.

Ogata, K., 2010. Ingeniería de control moderna, quinta Edición. Editorial Pearson Educación.

Pérez M. A., Pérez, A., Pérez, E., 2008. Introducción a sistemas de control y modelo matemático para sistemas lineales invariantes en el tiempo.

Pololu Robotics & Electronics. (17 de Septiembre de 2019). Obtenido de Pololu Corporation: https://www.pololu.com/product/1451/specs

Staticboards. (10 de Septiembre de 2019). Obtenido de https://www.staticboards.es

Valluru, S., Singh, M., Singh, S., 2016. Prototype design and analysis of controllers for one dimensional ball and beam system. In Power Electronics, Intelligent Control and Energy Systems (ICPEICES) 1, 1–6.

Ziegler, J., Nichols, N., 1942. Optimum settings for automatic controllers. Trans. ASME.

Apéndice A. Código Fuente

Se anexa el código fuente para la transmisión y recepción de datos entre la tarjeta Arduino Mega 2560 y el software Simulink-Matlab™. El lenguaje de programación está basado en C++ bajo el entorno de desarrollo Arduino.

```
#include "Servom_PI_Ext.h"
#include "rtwtypes.h"
#include <ext_work.h>
#include <ext_svr.h>
#include <ext_share.h>
#include <updown.h>
volatile int IsrOverrun = 0;
static boolean_T OverrunFlag = 0;
void rt_OneStep(void)
{
 /* Check for overrun. Protect OverrunFlag against
preemption */
  if (OverrunFlag++) {
   IsrOverrun = 1;
   OverrunFlag--;
   return;
  }
#ifndef _MW_ARDUINO_LOOP_
  sei();
```

```
#endif;
 Servom_PI_Ext_step();
 /* Get model outputs here */
#ifndef _MW_ARDUINO_LOOP_
 cli();
#endif;
 OverrunFlag--;
 rtExtModeCheckEndTrigger();
}
volatile boolean_T stopRequested = false;
int main(void)
{
 volatile boolean_T runModel = true;
 float modelBaseRate = 0.01;
 float systemClock = 0;
 init();
 MW_Arduino_Init();
 rtmSetErrorStatus(Servom_PI_Ext_M, 0);
 /* initialize external mode */
 rtParseArgsForExtMode(0, NULL);
 Servom_PI_Ext_initialize();
 sei();
 /* External mode */
rtSetTFinalForExtMode(&rtmGetTFinal(Servom_PI_Ext_
M));
 rtExtModeCheckInit(1);
 {
  boolean_T rtmStopReq = false;
  rtExtModeWaitForStartPkt(Servom_PI_Ext_M-
>extModeInfo, 1, &rtmStopReq);
  if (rtmStopReq) {
   rtmSetStopRequested(Servom_PI_Ext_M, true);
  }
 }
 rtERTExtModeStartMsg();
 cli();
 configureArduinoAVRTimer();
 runModel =
  (rtmGetErrorStatus(Servom_PI_Ext_M) == (NULL))
&& !rtmGetStopRequested
  (Servom_PI_Ext_M);
#ifndef _MW_ARDUINO_LOOP_
 sei();
#endif;
 sei();
 while (runModel) {
  /* External mode */
  {
   boolean_T rtmStopReq = false;
   rtExtModeOneStep(Servom_PI_Ext_M-
>extModeInfo, 1, &rtmStopReq);
   if (rtmStopReq) {
    rtmSetStopRequested(Servom_PI_Ext_M, true);
   }
  }
  stopRequested = !(
            (rtmGetErrorStatus(Servom_PI_Ext_M)
== (NULL)) &&
  runModel = !(stopRequested);
  runModel = runModel && MW_Arduino_Loop();
 }
 /* Disable rt_OneStep() here */
 /* Terminate model */
 Servom_PI_Ext_terminate();
 rtExtModeShutdown(1);
 cli();
 return 0;
}
```