# Metodología para la implementación de controlador difuso tipo Takagi-Sugeno en PLC s7-300

## *A method to implement a Takagi-Sugeno fuzzy controller using a PLC s7-300*

**Cristian Guarnizo Lemus**
Ingeniero electricista, magíster en Ingeniería Eléctrica. Investigador del Instituto Tecnológico Metropolitano de Medellín, Colombia. *cristianguarnizo@itm.edu.co*

**Clasificación del artículo: Investigación (Conciencias)**

**Fecha de recepción: 5 de marzo de 2011** **Fecha de aceptación: 30 de mayo de 2011**

***Palabras clave:*** Controlador lógico programable, IEC 1131-3, lógica difusa, texto estructurado.

***Key words:*** Programmable logic controller, IEC 1131-3, fuzzy logic, structured text.

**RESUMEN**

En este artículo se presenta una metodología para implementar controladores basados en lógica difusa en un PLC S7-300 empleando el lenguaje de programación SCL (Lenguaje de control estructurado) de STEP 7. Se presenta el diseño de la función difusa, declaración de variables, diseño de la evaluación de las funciones de pertenencia y reglas del sistema difuso. A partir de este esquema se pueden implementar modelos difusos más complejos, como adaptativos o autosintonizados. Se muestra un ejemplo de aplicación para un sistema de tiempo discreto simulado en el PLC, empleando un controlador difuso PI.

**ABSTRACT**

This paper presents a methodology for implementing fuzzy logic controllers based on a S7-300 PLC using the programming language SCL (Structured Control Language) in STEP 7. We present the design of the fuzzy function, variable declarations, and the evaluation design of membership functions and rules of the fuzzy system. Since this scheme can be implemented more complex fuzzy models, such as adaptive or Auto tuner. We present an application example for a discrete time simulated in the PLC using a PI fuzzy controller.

* * *

## 1. INTRODUCCIÓN

Actualmente el controlador lógico programable (PLC) es utilizado en la industria para realizar control de sistemas basados en eventos y para el control de procesos continuos por medio de procedimientos en tiempo discreto. Esto se debe a las capacidades que tiene el PLC de procesar señales análogas y digitales. A partir de esto se ha desprendido una investigación entre las limitaciones y la capacidad que tiene el PLC para realizar control discreto en un tiempo lo suficientemente corto (tiempo real). Esto con el fin de poder implementar controladores avanzados para procesos complejos en la industria, aprovechándose los equipos instalados, sin necesidad de requerir de un controlador especial.

En [1-6] se han hecho propuestas y validación de métodos basados en lógica difusa implementados en el PLC, en cada uno se comparan los métodos difusos contra los convencionales (PID), donde los primeros presentan un mejor desempeño al controlar los diferentes sistemas analizados. En los documentos anteriores solo se muestran los resultados, pero no, la forma en que se implementaron dichos sistemas difusos, excepto en [4], donde se describe de forma detallada cómo implementar el controlador difuso en la plataforma del TSX 21-37, con el inconveniente de requerir de un *plug-in* especial de propiedad de la empresa Telemecanique, para poder implementar el controlador difuso.

En este documento se especifica la forma de implementar un controlador difuso tipo Takagi-Sugeno en un PLC S7-300, la elección del esquema Takagi-Sugeno se debe a su menor costo computacional comparado con el sistema difuso tipo Mandami; pero si en el diseño inicial se dispone de un sistema difuso tipo Mandami, éste se puede convertir a Sugeno, como se describe en [7], dicho proceso consiste en aproximar el modelo Mandami a partir de mínimos cuadrados o algoritmos evolutivos. De esta manera cualquier sistema de inferencia o control difuso se puede implementar por medio de un Takagi-Sugeno con un costo computacional menor.

Si se desea reducir considerablemente el costo computacional y mejorar el desempeño del controlador difuso se pueden emplear técnicas evolutivas que decidan cuáles son las mejores reglas, la cantidad de funciones de pertenencia y los parámetros de las funciones de membrecía de las entradas y salidas, con el objetivo de reducir el error entre la referencia y la salida del sistema, como algoritmos genéticos [8], colonia de hormigas [9] y cúmulo de partículas [10].

## 2. METODOLOGÍA

En general el modelo difuso más sencillo de implementar es el tipo Takagi-Sugeno, debido a que el cálculo de la salida tiene un costo computacional mucho menor que el modelo Mandami [11]. Los pasos de aplicación de un modelo difuso son: fusificación de las entradas, evaluar las funciones de pertenecía de cada entrada de acuerdo con el valor fusificado, evaluación de las reglas, cálculo y defusificación de la salida. Los pasos de fusificación y defusificación consisten en llevar los valores de entrada y salida a una escala definida en el sistema difuso. A continuación, se describe la forma de programar cada paso necesario para evaluar un sistema difuso en general.

Para desarrollar un controlador difuso adaptativo en el cual los parámetros se sintonizan de acuerdo con el error, se debe crear un bloque de función, el cual permite guardar valores y retenerlos en cada ciclo de ejecución. Al almacenar de esta forma las posiciones de las funciones de pertenencia de la entrada y la salida se pueden desarrollar algorit-

mos adaptativos. Pero si se requiere un modelo estático, en el cual los parámetros que definen el modelo no se reajustan, entonces se puede realizar todo el proceso en una función, donde la salida de la función solo depende de los valores de las entradas. El lenguaje de programación de texto estructurado (ST) del estándar IEC 6113-3[12] es similar al lenguaje de programación Pascal, el lenguaje SCL (*Structured Control Language*) de STEP 7 se basa en ST. El diseño del sistema difuso se realiza en SCL.

La declaración de la función difusa y sus variables en sintaxis SCL de STEP 7, para 2 entradas, cada una con 3 funciones de pertenencia, y una salida del sistema difuso, es:

```
FUNCTION FC10: REAL
VAR_INPUT
 U1,U2: REAL; //Entradas del sistema
END_VAR

VAR_TEMP
EFP: ARRAY [1..2,1..3] OF REAL;
AFP: ARRAY [1..2,1..3] OF BOOL;
WS: ARRAY [1..3] OF REAL;
S: ARRAY [1..3] OF REAL;
num, den, temp: REAL;
 n: INT;
END_VAR
```

Las dimensiones de los vectores **EFP** y **AFP** dependen de la cantidad de entradas (filas) y el número de funciones de pertenencia (columnas). El vector **EFP** almacena los valores de la evaluación de las funciones de pertenencia, mientras que **AFP** almacena qué funciones de pertenencia se activaron (valor lógico verdadero sí se activó y falso de lo contrario). El vector **AFP** facilita posteriormente la evaluación de las reglas del sistema difuso. El vector **S** contiene las posiciones de los *singletons* (constantes) de las funciones de la salida. El vector **WS** contiene los pesos de cada una de las salidas. La variable *temp* almacena valores intermedios en la asignación de los valores de **WS**. La variable n se emplea para realizar el ciclo iterativo FOR para el cálculo de la salida. Finalmente, *num* y *den* son las variables del numerador y denominador para el cálculo de la salida, respectivamente.

### 2.1. Definición y evaluación de las funciones de pertenencia

Las funciones de pertenencia recomendadas son del tipo triangular y trapezoidal, debido a su bajo costo computacional. Para evaluar las funciones de pertenencia de una entrada cualquiera, se emplean estructuras condicionales IF. A continuación se muestra un ejemplo con una función triangular:

```
IF (U1>15,0) AND (U1<30,0) THEN
AFP[1,2]:=True;
 IF U1<=25,0 THEN
EFP[1,2]:=(U1-15,0)/10,0;
 ELSE
EFP[1,2]:=-(U1-30,0)/5,0;
END_IF;
END_IF;
```

Si la entrada se encuentra dentro del rango de definición de la función de pertenencia, entonces se hace verdadero el valor en el vector de activaciones (**AFP**) correspondiente a la función de

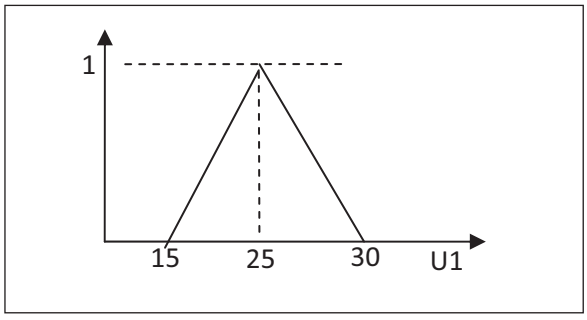


**Fig. 1.** Función triangular.

pertenencia, en este caso es la columna 2. Posteriormente, dependiendo del valor de la entrada se calcula el valor de la función de pertenencia y se almacena en **EFP**. Se debe reducir al máximo la cantidad de operaciones que se realizan al evaluar la función de pertenencia.

### 2.2. Métodos de implicación

Antes de definir las reglas, se plantea cuáles son los métodos de implicación entre las funciones de pertenencia de las entradas. Los métodos de implicación unen los valores fusificados de las entradas para generar las reglas y posteriormente la salida. En general las reglas tienen la siguiente forma:

SI U1 es FP1 Y U2 es FP1 OR U1 es FP2 Y U2 es FP1 ENTONCESDU es **S** [1]

Las uniones Y generalmente se aplican seleccionado el mínimo de los valores de pertenencia obtenidos al evaluar las entradas, mientras que en las uniones O se aplica el máximo entre los mínimos obtenidos por la uniones Y. Las uniones Y y O se usan para definir las reglas difusas. Al final de las reglas, el consecuente está asociado con una función de pertenencia de la salida.

### 2.3 Definición de las reglas

Para evaluar las reglas se emplea el vector booleano de activación de las funciones de pertenencia (AFP), de esta manera se hace más fácil y rápido evaluar cada regla:

```
IF AFP[1,1] AND AFP[2,1] THEN
 IF EFP[1,1]>EFP[2,1] THEN
        temp:=EFP[2,1];
 ELSE
        temp:=EFP[1,1];
 END_IF;
IF WS[1]<temp THEN
        WS[1]:=temp;
END_IF;
END_IF;
```

Cada vez que se evalúa una regla, inicialmente se almacena el mínimo entre los valores de las funciones de pertenencia que hacen parte de la regla. Posteriormente, si existen varias reglas asociadas a una misma salida, entonces se compara el valor obtenido en la regla con el valor del peso almacenado previamente a la salida de dicha regla. Se asigna como peso el mayor entre los dos.

### 2.4 Calculo de la salida

La salida del sistema difuso tipo Takagi-Sugeno se calcula de la siguiente forma:

$$Salida = \frac{\sum_{i=1}^{n} \boldsymbol{WS}[n]\boldsymbol{S}[n]}{\sum_{i=1}^{n} \boldsymbol{WS}[n]} \qquad (1)$$

Donde n es el número de funciones de pertenencia de la salida, **WS**[n] contiene el máximo entre las reglas que comparten la salida **S**[n]. El programa en SCL de la Ec. (1) para 3 funciones de salida, es:

```
FOR n:=1 TO 3 DO
        num:=WS[n]*S[n]+num;
        den:=WS[n]+den;
END_FOR

FC10:=num/den;
```

donde finalmente, FC10 devuelve el valor de la salida del controlador difuso. Recordar que num y den se deben inicializar con 0.

Con los pasos desarrollados en las secciones previas se puede crear el bloque de función que realiza la evaluación del sistema difuso, como se muestra a continuación.

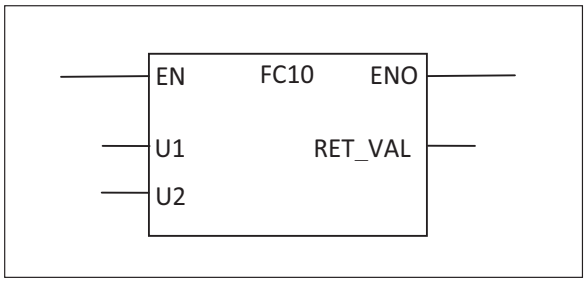


**Fig. 2.** Función difusa en forma de bloque del estándar.

### 2.5 Bloque difuso en Step 7

El bloque de función finalmente diseñado se puede invocar desde cualquier otro lenguaje establecido en el estándar IEC 1131-3 ([12] Lewis, 1998). A continuación se muestra el bloque del sistema de inferencia difuso diseñado:

Como se observa en la Fig. 2, el bloque cuenta con las entradas U1 y U2, y la salida RET_VAL, que es la salida del controlador difuso. La entrada EN, es usada para habilitar el uso del bloque, y la salida ENO indica por medio de un valor lógico si se ejecutó correctamente el bloque.

## 3. RESULTADOS

El sistema lineal de tercer orden por controlar es simulado en el PLC por medio de una función. El sistema está dado por la siguiente función de transferencia [13]:

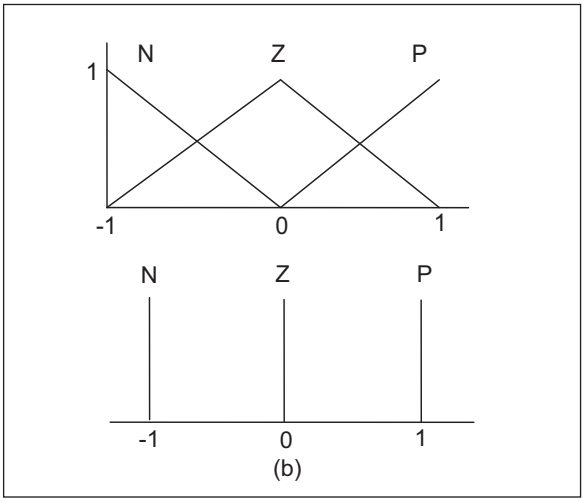


**Fig. 4.** Funciones de pertenencia (a) de las entradas (e(k) y Δe(k)), y (b) de la salida (Δu(k)).

$$\frac{U(s)}{R(s)} = \frac{5}{s^3 + 4,5s^2 + 5,5s + 15} \qquad (2)$$

El esquema de control empleado para el sistema dado en la Ec. (2), se basa en el modelo difuso PI propuesto en [1,14], mostrado en la Fig. 3, donde las entradas discretas del sistema difuso son el error e(k) y cambio en el error Δe(k).

Donde r(k) es la referencia, y(k) es la salida de la planta y u(k) es la entrada de la planta en el instante k. Las constantes *Ke*, *Kde* y *Ku*, se utilizan para sintonizar el proceso de control. Las funcio-

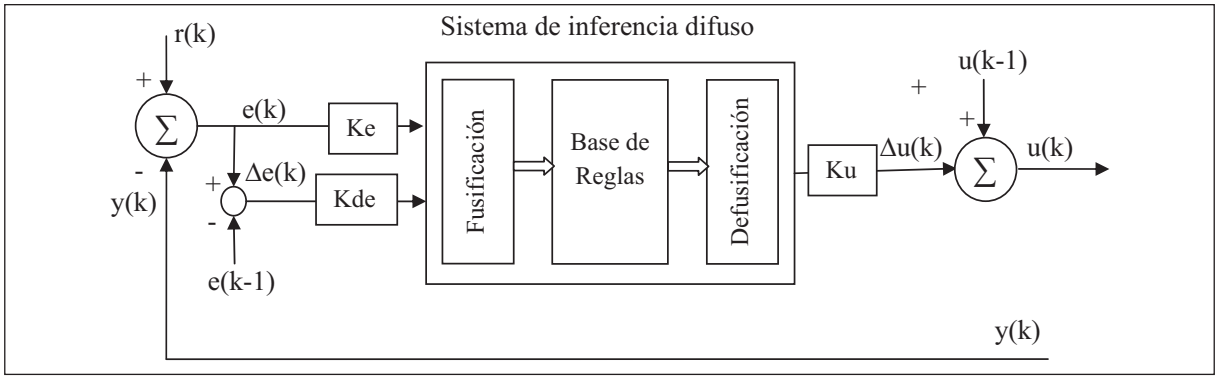


**Fig. 3.** Esquema del controlador difuso implementado.

nes de pertenencia en la entrada y la salida del sistema difuso están definidas como se muestra en la Fig. 4.

Se programa la función difusa con la metodología anteriormente descrita, se discretiza el sistema dado en Ec. (2) con un tiempo de muestreo igual a 0,2s. Posteriormente se implementa dicho sistema en PLC por medio de un bloque de función. Los datos de la salida de la planta son almacenados en un bloque de datos. El controlador difuso se agrega al bloque de organización OB35 que permite realizar control discreto de sistemas continuos. Las reglas empleadas para el controlador difuso se definen en la tabla 1.

**Tabla. 1.** Función triangular. Definición de la base de reglas.

| e(k)/Δe(k) | N | Z | P |
|---|---|---|---|
| N | Z | P | P |
| Z | N | Z | P |
| P | N | N | Z |

Posteriormente, se realizan dos pruebas variando las constantes para observar el desempeño del controlador. Para la configuración inicial se asignan los parámetros *Ke*=1, *Kde*=1y*Ku*=0,2, con los cuales se obtuvo la respuesta mostrada en la Fig. 5.

Para estos valores se observa que la salida del sistema presenta oscilaciones alrededor de la referencia. Se procede entonces a variar las constantes para obtener un mejor desempeño. Sintonizando manualmente los parámetros *Ke*, *Kde* y *Ku*, se obtuvo la siguiente respuesta al escalón con los parámetros *Ke*=2, *Kde*=0,01 y *Ku*=0,1:

En la Fig. 6 se presenta una respuesta en la cual la salida del sistema llega a la referencia de una manera suave. El controlador difuso implementado es altamente sensible a los valores de las constantes *Ke*, *Kde*, *Ku*. El tiempo de ejecución estimado por el Software Step 7 se puede observar en la Fig. 7.

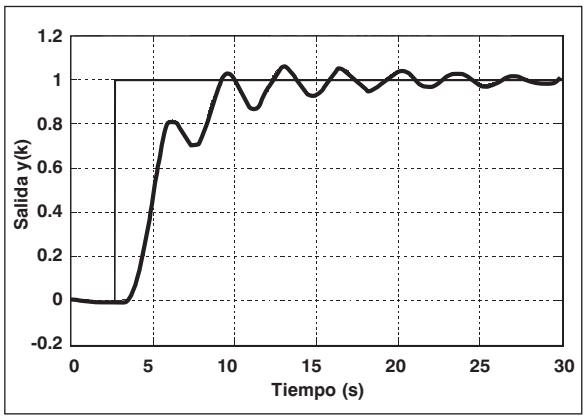


**Fig. 5.** Respuesta al escalón para el controlador difuso con Ke=1, Kde=1, Ku=0,2.

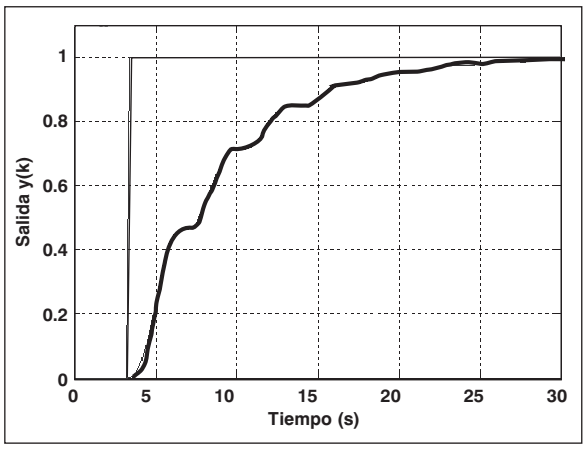


**Fig. 6.** Respuesta al escalón para el controlador difuso con Ke=2, Kde=0,01, Ku=0,1.

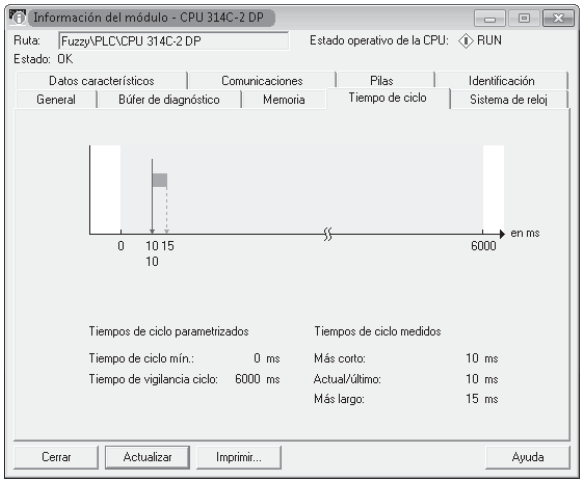


**Fig. 7.** Tiempos de ciclo del controlador difuso.

El tiempo de cálculo del controlador difuso, además del cálculo del sistema simulado está entre 10 y 15 ms. Se puede realizar control en tiempo real empleando técnicas difusas implementadas en el PLC, se debe considerar que el tiempo de ciclo, fue cálculo para 3 funciones de pertenencia por entrada, la complejidad del sistema difuso crece exponencialmente a partir de la cantidad de funciones de pertenencia que se asignen por entrada.

## 4. DISCUSIÓN

La metodología de programación se basa en lenguaje estructurado definido en el estándar IEC 61131-3, por tal razón, cualquier PLC que cumpla con este estándar, puede realizar la metodología anteriormente propuesta. En este mismo estándar en la sección 7, se define un lenguaje de programación de lógica difusa, llamado FCL (*Fuzzy control language*). Pero actualmente los PLC solo cumplen con los lenguajes definidos en el numeral 3, por esta razón, si se requiere diseñar un controlador difuso se debe programar de una forma similar descrita en este documento.

Si no se realiza una selección adecuada de los parámetros *Ke*, *Kde* y *Ku* se puede obtener una respuesta como en el resultado mostrado en la Fig. 5. Los sistemas difusos son altamente sensibles a la sintonización de todos sus parámetros para obtener un desempeño aceptable. La selección se puede realizar por medio de técnicas evolutivas como la presentada en [10].

A partir de la metodología aquí descrita, se pueden implementar controladores difusos combinados con PID, Control predictivo (Sistema difuso que estime el modelo del sistema), sistemas difusos autosintonizados.

## 5. CONCLUSIONES

Se puede implementar un controlador basado en Lógica difusa en un PLC, para realizar control de sistemas lineales o no lineales; adicionalmente, se puede controlar un proceso de eventos discretos. Con la metodología descrita se pueden realizar otras topologías o esquemas de control difuso.

Los parámetros de configuración del sistema difuso se deben seleccionar adecuadamente por medio de simulaciones para obtener un desempeño aceptable del controlador.

El PLC como herramienta industrial puede ser utilizado para desarrollar sistemas de control avanzados.

## 6. FINANCIAMIENTO

Este trabajo está enmarcado bajo el proyecto P-09235 registrado y financiado por el Centro de Investigaciones del Instituto Tecnológico Metropolitano, Medellín, Colombia.

## REFERENCIAS


[1] H. Li and S. Tso, "A fuzzy PLC with gain-scheduling control resolution for a thermal process - a case study," *Control Engineering Practice*, vol. 7, pp. 523-529, 1999.

[2] O. Karasakal, E. Yesil, M. Guzelkaya and I. Eksin, "The implementation and comparison of different type self-tuning algorithms of fuzzy PID controllers on PLC," *World Automation Congress*, Sevilla, Jul. 2004.

[3] O. Karasakal, E. Yesil, M. Guzelkaya and I. Eksin, "Implementation of a New Self-Tuning Fuzzy PID Controller on PLC," *Turk. J. Electrical Engeenering & Computer Science*, vol. 13, no. 2, pp. 277-286, 2005.

[4] H. Ferdinando, "The Implementation of low cost fuzzy logic controller for PLC TSX 37-21," *International Conference on Intelligent and Advanced Systems 2007*, Kuala Lumpur, Malaysia, Nov. 2007.

[5] M. Yahyaei, J.E. Jam and R. Hosnavi, "Controlling the navigation of automatic guided vehicle (AGV) using integrated fuzzy logic controller with programmable logic controller (IFLPLC)—stage 1," *The International Journal of Advanced Manufacturing Technology*, vol. 47, no. 5-8, pp. 795 - 808, 2010.

[6] X. Liu, J. Geng, S. Teng and C. Li, "A fuzzy-PID controller with adjustable factor based on S7-300 PLC," *International Journal of Modelling, Identification and Control 2009*, vol. 7, no. 4, pp. 371 – 375, 2009.

[7] J. Jassbi, S.H. Alavi, P.J. Serra and R. Ribeiro, "Transformation of a Mamdani FIS to first order sugeno FIS," *IEEE international conference on Fuzzy systems*, London, United Kingdom, Jul. 2007.

[8] P. Siarry and F. Guely, "A genetic algorithm for optimizing Takagi-Sugeno fuzzy rule bases," *Fuzzy Sets and Systems*, vol. 99, no. 1, pp. 37-47, Oct. 1998.

[9] R. Martínez-Marroquín, O. Castillo and J. Soria, "Optimization of membership functions of a fuzzy logic controller for an autonomous wheeled mobile robot using ant colony optimization," *Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control*, Springer Berlin / Heidelberg, pp. 3-16, 2009.

[10] E. Omizegba and F. Monikang, "Fuzzy logic controller parameter tuning using particle swarm algorithm," *International Journal of Modelling, Identification and Control 2009*, vol. 6, no.1, pp. 26 – 31, 2009.

[11] International Electrotechnical Commission (IEC), *IEC 1131 Programmable Controllers. Part 7: Fuzzy Control Programming*. IEC 1131-7, Jan. 1997.

[12] R.W. Lewis, *Programming Industrial Control Systems Using IEC 1131-3*. The Institution of Engineering and Technology, London: United Kingdom 1998.

[13] H. Li, "A comparative design and tuning for conventional fuzzy control," *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, vol. 27, no. 5, pp. 884-889, Oct. 1997.

[14] I. Altas and A. Sharaf, "A generalized direct approach for designing fuzzy logic controllers in matlab/simulink GUI environment," *International Journal of Information Technology and Intelligent Computing*, vol. 1, no.4, 2007.