Self-Tuning Fuzzy Looper Control for Rolling Mills

F. Janabi-Sharifi
Department of Mechanical Engineering
Ryerson Polytechnic University
Toronto, Ontario, Canada M5B 2K3

J. Fan
Quad Engineering Inc.
North York, Ontario, Canada M3B 2R2

Abstract

This paper deals with the problem of looper control for tension-free rolling. Conventional controllers cannot deal effectively with unmodeled dynamics and large variations which can lead to scrap runs and damages to machinery. Therefore, a fuzzy controller has been designed to use the expert knowledge of the operators for disturbed process control. Also, a self-tuning algorithm is incorporated for both on-line and off-line tuning of the fuzzy membership functions. This paper discusses the design of the fuzzy logic controller and its self-tuning. The effects of various design options are discussed and practical conclusions are made. Results from simulations are also presented.

1 Introduction

The loop control methods are commonly used for flexible cross-sections at the intermediate and finishing submills. These methods rely on an initial formation of a bar loop by utilizing mechanical deflectors and proper motor speed adjustments (Fig. 1). In fact, each stand roll speed has to be synchronized to the speed of the bar exiting the previous stand. Otherwise, *push* or *pull* conditions would occur and loop might be formed. The height of the formed loop could serve as a tension indicator. Maintaining a constant desired loop height will be done by adjusting the motor speed ratios and will indicate no tension/compression status. The height of the loop (H) is measured and compared to reference height (H_0, indicating zero tension condition) to obtain a correction command for motor speed control unit. An example is the implementation of the looper control in a 17 stand bar mill in the Chapparel Steel mill in Milwaukee by ASEA [7]. A characteristic control problem is the introduction of the disturbances of different nature. These disturbances are caused by the increasing use of low temperature heating of slabs, high-carbon steels, and high-speed rolling. Due to increasing demand for high precision in thickness, width, and crown of the strips, it is very important to develop a high-performance and robust looper control. In this paper, the potential of fuzzy controllers will be examined and the issues on the application of fuzzy controllers for

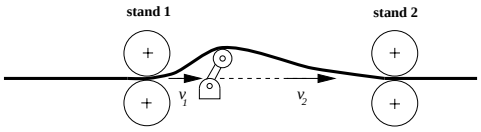

Figure 1: The rolling mill with looper control.

looper control will be discussed. We were motivated to explore the application of fuzzy looper control for the following reasons.
(i) Conventional non-interactive controllers (e.g. PD, and PID types) fail to meet high precision and disturbance robustness requirements, mainly due to the presence of large disturbances. Therefore there is usually a trial period for the start of the production. Fuzzy controllers prove to be relatively robust to noises.
(ii) In rolling mill applications, it is very difficult or costly to use conventional model based approaches which require a known mathematical model of the system. Also many existing adaptive control methods deal only with linear systems. A fuzzy control design does not require a formal model of the problem.
(iii) Obtaining training data, e.g., for a neural-based controller, might be expensive. A fuzzy controller does not initially require a training set.
(iv) Usually sufficient knowledge in the form of linguistic if-then rules is available by interviewing a human operator.
An alternative would be to use artificial neural net-based adaptive controllers for unknown nonlinear systems [5], [6]. However, these controllers cannot incorporate linguistic control rules or system descriptions directly into the controllers. In rolling mills, there is a lot of human linguistic knowledge and incorporating this knowledge into controllers in a systematic way is very important.

Despite the enumerated advantages, a few problems might arise when implementing a fuzzy logic controller (FLC). The translation of good linguistic rules depends on the knowledge of the control expert. In many cases, redundant or insufficient rules might be specified. Also, translation into fuzzy set theory is not formalized yet and arbitrary choices concerning the shape or membership functions or T-operators might be made. These uncertainties in the design of a FLC usually results in

a heuristic tuning to overcome the initial design errors [4]. Therefore, it is very important to have learning procedures which can tune the system automatically. The automated FLC tuning was first proposed by Procyk and Mamdani [10]. Most of the work in automated FLC tuning has focused on neural nets [1], [3], [8]. However, these approaches lack sufficient generalization and expressing capability of the acquired knowledge. Furthermore, our experiments indicated long training effort for these approaches.

In this paper, we will implement a self-tuning method based on descent technique for tuning membership functions of FLC. Both off-line and on-line tuning will be considered. This will contribute to a rapidly tunable FLC framework for rolling mills. To the best knowledge of the authors, this is the first on-line tunable FLC framework for the looper control in rolling mills. Different design issues will be discussed and practical conclusions will be derived.

2 Looper System Model

The plant is a rolling mill with a looper for tension control (Fig. 1). The tension is caused by the difference between the exit speed v_1 in stand one and the entry speed v_2 in stand two. The speed difference results in the storage of the strip length, which can be obtained from the integral of speed difference:

$$L(t) = \int_t [v_1(t) - v_2(t)]dt \tag{1}$$

and the looper height is related to the storage length by

$$H(t) = \frac{1}{\alpha + \beta/\sqrt{L(t)}}. \tag{2}$$

Here α and β depend on the looper parameters and are determined experimentally. For the simulation purposes, we adopted $\alpha = 0.0001955567$ and $\beta = 0.028845145$ for $L(t)$ in mm. The Jacobian of the plant is then given by

$$J_{\Delta v_1} = \frac{\partial H(t)}{\partial(\Delta v_1)} = \frac{\beta(v_2(t) - v_1(t))}{2\sqrt{L(t)}(\alpha\sqrt{L(t)} + \beta)^2 \dot{v_1}(t)}. \tag{3}$$

3 Design of Fuzzy Controller

The control structure of the looper system is shown in Fig. 2. In this section, we will not consider any tuning aspect and the focus will be on FLC. The process control can be described as follows:

$$x(k) = [x_1(k), x_2(k)]^T = [e(k)/K_{in1}, \Delta e(k)/K_{in2}]^T, \tag{4}$$

$$e(k) = y(k) - y_r(k), \quad \Delta e(k) = e(k) - e(k-1), \tag{5}$$

$$y(k) = H(k), \quad y_r(k) = H_r(k) \tag{6}$$

$$u(k) = \Delta v_1(k) = U[x_1(k), x_2(k)]. \tag{7}$$

Here state vector x includes error $e(k)$ and error change $\Delta e(k)$ as the the inputs to the controller, and the controller output $u(k)$ is generated using $U[x_1(k), x_2(k)]$, nonlinear mapping implemented using fuzzy logic. The input scaling factors are K_{in1} and K_{in2} respectively. Also, $y(k)$ and $y_r(k)$ are system output and reference command respectively. In the following we describe how $U[x_1(k), x_2(k)]$ is implemented. The realization

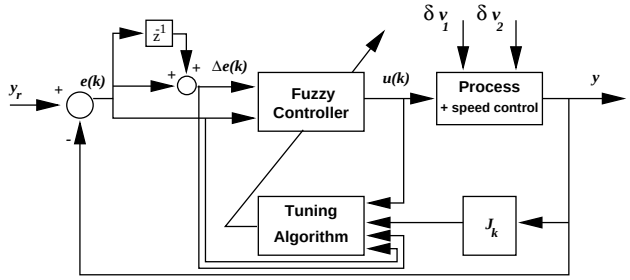

Figure 2: The structure of the lopper control system.

of the function $U[x_1(k), x_2(k)]$ is based on a fuzzy logic method and consists of three stages: fuzzification, decision making fuzzy logic, and defuzzification. The process of fuzzification transforms the inputs $x_1(k)$ and $x_2(k)$ into the setting of linguistic variables which maybe viewed as labels of a fuzzy set. In this work, the following linguistic variables were used for e, and Δe respectively:
$L_{x_1} = \{\mathbf{NB}, \mathbf{NS}, \mathbf{PS}, \mathbf{PB}\}$, $L_{x_2} = \{\mathbf{N}, \mathbf{Z}, \mathbf{P}\}$.
Here the meaning of each variable should be clear from its mnemonic: **NB** (Negative Big), **NS** (Negative Small), **N** (Negative), **Z** (Zero), **P** (Positive), **PS** (Positive Small), and **PB** (Positive Big). The membership functions of the inputs to the controller (Fig. 3) were all assumed to be triangular. The fuzzy partitioning of the Fig. 3 has been based on normalization and choosing scaling factors by inspecting the operation range. The scaling factor is based on the expert knowledge and can be rapidly adjusted by means of a few trials. For each input x_1 and x_2, we assign numbers $\mu_{A_{i1}}(x_1)$ and $\mu_{A_{i2}}(x_2)$ using membership functions A_{i1} and A_{i2} associated with L_{x_1} and L_{x_2} respectively:

$$\mu_{A_{i1}}(x_1) = A_{i1}(x_1(k)), \mu_{A_{i2}}(x_2) = A_{i2}(x_2(k)). \tag{8}$$

These numbers will be used for fuzzy decision making which is described next. Associated with the fuzzy logic decision process is a set of fuzzy rules $R = \{R_1, R_2, R_3, \cdots, R_{12}\}$. The rules are given in Fig. 4, where the values in brackets indicate Mamdani rules. The singleton rule will have a form of:
Rule i: **If** x_1 is l_{i1} **and** x_2 is l_{i2}, **then** $u = w_i$.
Here $l_{i1} \in L_{x_1}$, and $l_{i2} \in L_{x_2}$ are set of linguistic terms attached to x_1, and x_2 respectively. Also, w_i is a real number associated with the singleton controller which is a special case of Takagi-Sugeno controller [12] with all coefficients of higher order equal to zero. Fuzzy rules were developed heuristically to facilitate the shaping

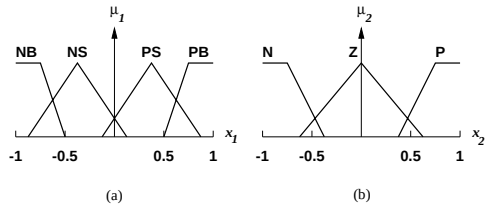

Figure 3: Membership functions for the state variables: (a) $x_1 = e$, (b) $x_2 = \Delta e$. The domains for both variables have been normalized to [-1, 1].

Δe \ e	NB	NS	PS	PB
N	w_1 (PB)	w_2 (PM)	w_3 (PS)	w_4 (PZ)
Z	w_5 (PS)	w_6 (PZ)	w_7 (NZ)	w_8 (NS)
P	w_9 (NZ)	w_{10} (NS)	w_{11} (NM)	w_{12} (NB)

Figure 4: Fuzzy rule-base.

of the looper's response and to simplify tuning of the controller. If a different response was desired for a particular range of input variables, only a few fuzzy rules would need to be altered. The ability to modify the controlled response locally, while not significantly altering the global response, is an attractive feature of fuzzy control, in particular from learning point of view. The isosceles triangles used for the membership functions could be characterized by their center and width. For instance, membership function A_{ij} could be expressed by its center a_{ij} and the width b_{ij} (Fig. 5). Therefore, the grade of membership for A_{ij}, denoted by $\mu_{A_{ij}}$, could be obtained from

$$\mu_{A_{ij}}(x_j) = 1 - \frac{2 \mid x_j - a_{ij} \mid}{b_{ij}}, \qquad j = 1, 2. \tag{9}$$

Each relevant rule, given a pair of $e(k)$ and $\Delta e(k)$, first assigns $\mu_{A_{i1}}(x_1(k))$ and $\mu_{A_{i2}}(x_2(k))$ using (8) and then assigns to it a value using the production *t-norm* defined as follow:

$$\mu_i(x_1(k), x_2(k)) = \mu_{A_{i1}}(x_1(k)) \cdot \mu_{A_{i2}}(x_2(k)) \tag{10}$$

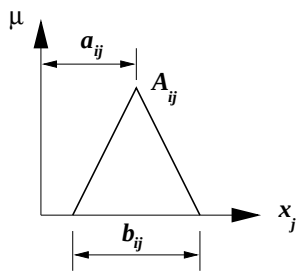

Figure 5: A parameterized membership function.

for the ith rule. This process is repeated for all the relevant rules.

Basically, defuzzification is a mapping from a space of fuzzy control action defined over a universe of discourse into a space of non-fuzzy control actions. Several methods of defuzzification can be considered. The defuzzification process considered here takes a form of Takagi-Sugeno output value computation method, given by:

$$\begin{aligned} u(k) &= U[x_1(k), x_2(k)] \\ &= K_{out} \frac{\sum_{i=1}^{n} \mu_i(x_1(k), x_2(k)) w_i}{\sum_{i=1}^{n} \mu_i(x_1(k), x_2(k))}, \end{aligned} \tag{11}$$

where K_{out} is a scaling factor, and w_i is a real number associated with the consequent part of the rule (or the center of relevant membership function of the output in Mamdani rule-base).

4 Tuning Algorithm

Tuning FLC is a very difficult task as it has more parameters to be tuned than its non-fuzzy counterparts. It is possible to tune rules, operators, and/or membership functions (MFs). In this work, we will focus on MF tuning for better performance. We initially focused on conventional multilayer perceptron using backpropagation algorithm [11] for tuning MFs. However, the results were not encouraging with respect to convergence and speed of tuning. Therefore, we focused on descent methods [9]. The tuning algorithm presented here is relatively simple and fast for on-line tuning. The objective function to be minimized is defined as:

$$E(k) = \frac{1}{2} \sum_{k} [y(k) - y_r]^2 \tag{12}$$

with $y(k)$ as the system output in the kth instance. It can be easily shown that the learning rules can be expressed as:

$$a_{ij}(k+1) = a_{ij}(k) - k_a \frac{\partial E(k)}{\partial a_{ij}(k)} \tag{13}$$

$$b_{ij}(k+1) = b_{ij}(k) - k_b \frac{\partial E(k)}{\partial b_{ij}(k)} \tag{14}$$

$$w_i(k+1) = w_i(k) - k_w \frac{\partial E(k)}{\partial w_i(k)} \tag{15}$$

where k_a, k_b, and k_w are learning coefficients. The gradients in equations (13), (14), and (15) can be derived from equations (9) to (11) [2]. The gradients will include $J_k = (\frac{\partial H}{\partial u})_k$ which is the Jacobian. The summation $\sum_k$ will appear in the gradients. The summation is for off-line learning which uses the summation of measurements at the discrete instances after the end of control cycle. No summation $\sum_k$ is required in the

above equations for on-line learning. In our simulations, we will use discrete linearized Jacobian instead of (3), defined as

$$J_k = \frac{y(k) - y(k-1)}{u(k) - u(k-1)} \tag{16}$$

or its sgn defined by:

$$\text{sgn}(J_k) = \begin{cases} 1 & \text{if } J_k > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

The training procedure works as follows:

1. Initiate the inference rules and membership functions.
2. Calculate y, x_1, and x_2 using (2) and (4)-(6).
3. Calculate Jacobian using (3), or (16), or (17).
4. Update the membership functions using (12).
5. Calculate the control output using (7). Go to Step 2.

5 Simulation Results and Discussion

Simulations were run to verify the effectiveness of the proposed method. The simulation parameters were chosen to be: $v_1(0) = 7000$ mm/s, $v_2 = 7000$ mm/s, $H_r = 200$ mm, $\Delta H(0) = 50$ mm, and $\Delta T = 0.05$ s. The learning coefficients were determined after experiments: $k_a = k_b = 0.1$, $k_w = 0.01$ for off-line tuning, and $k_a = k_b = 0.002$, $k_w = 0.005$ for on-line tuning. The scaling factors were chosen to be: $K_{in1} = 60$, $K_{in2} = 10$, and $K_{out} = 50$. The results are shown in Figs. 6 to 12. The dotted and solid lines represent the responses of fuzzy logic control without tuning (FLC) and with tuning (TFLC) respectively. The initial values of [a, b] for the membership functions of e were: NB: [-1.00, 1.20]; NS: [-0.40, 1.00]; PS: [0.40, 1.00]; PB: [1.00, 1.20]; and those for Δe were: N: [-0.95, 2.00]; Z: [0.00, 2.00]; P: [0.95, 2.00]. Also the initial values of output singletons (w_1 - w_{12}) were: [1.00, 0.70, 0.55, 0.3.0, 0.55, 0.30, -0.10, -0.35, -0.10, -0.35, -0.70, -1.00]. Several cases were studied under different conditions
(C1) Off-line, Jacobian model.
(C2) On-line, Jacobian model.
(C3) On-line, linearized Jacobian model.
(C4) On-line, Jacobian sign model.
Mamdani-based control was compared with Singleton-based control and it was shown that there was no obvious difference between the performances of the above controls. In the following part, only the results of Singleton-based control will be presented for the sake of brevity.

Off-line vs on-line learning. For the off-line learning, the parameters of the membership functions are tuned after each control cycle, while in on-line learning, the parameters are tuned after each time step. The results [2] indicated lower errors when the control was tuned. Also, it was shown that off-line learning provided lower undershoots than on-line learning.

Effect of plant Jacobian. The plant Jacobian affects the tuning algorithm. Due to the difficulty of providing true Jacobian, approximations of Jacobian are used. A linearized model of the plant Jacobian (16) could provide an easy-to-calculate approximation. Also, in order to improve the stability of the system, we examined the sign of linearized Jacobian (17) for parameter tuning. The results are shown in Figs. 6 to 8. For (C4), tuned values of [a, b] for NS, PS, and PB of e were: [-0.39, 1.00], [0.35, 0.99], and [0.96, 1.21] respectively. Also, those for N and Z of Δe were: [-0.98, 1.98] and [-0.01, 2.00] respectively. The changes in w values were more significant, i.e. w_3 - w_8 were tuned to: -0.40, 0.17, 0.55, 0.29, -0.25, and -0.58 respectively. It was concluded that the sign of linearized Jacobian provided better performance. Therefore, the sign function was used for the rest of simulations.

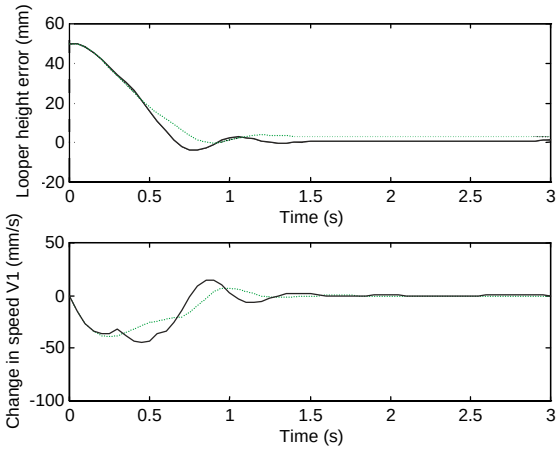

Figure 6: Comparison of step responses of FLC (dotted) and TFLC (solid) under (C2).

Disturbance rejection. To study the effectiveness of the proposed system in disturbance rejection, a step disturbance in setting of exit speed of strip in stand one (v_1) was introduced at $t = 3$ sec. The results [2] indicated that the tuned controller was capable of canceling the drop caused by the disturbance.

Effect of input and output scaling. As it is shown in Fig. 3, the input values must be mapped to the range of [-1, 1]. Therefore, the inputs e and Δe are scaled down by dividing them to K_{in1} and K_{in2} respectively. Also, the control output is scaled by multiplying it to K_{out}. Scaling factors have effects on the dynamic response. The response time decreases with the increase in K_{out} and/or decrease of K_{in} (Fig. 9). However, if K_{in} becomes too small, the system will be unsteady.

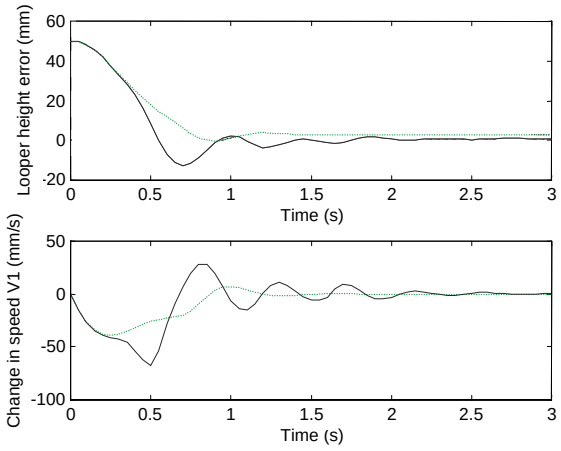

Figure 7: Comparison of step responses of FLC (dotted) and TFLC (solid) under (C3).

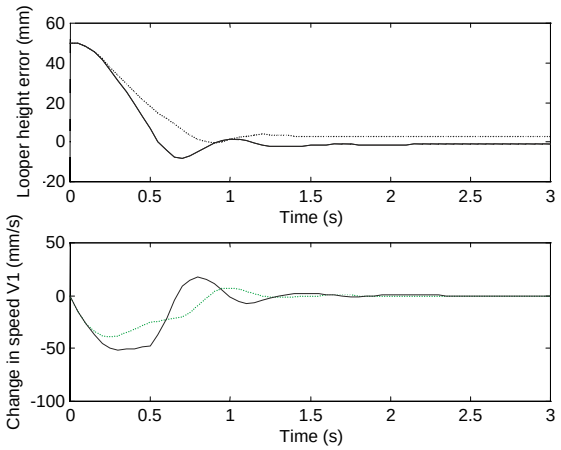

Figure 8: Comparison of step responses of FLC (dotted) and TFLC (solid) under (C4).

Fuzzy control vs PI/PID control. Comparisons with commonly used PI amd PID control laws were made. The PI/PID gains have been tuned to give optimum response (Figs. 10-12): $K_p = K_d = 5$, $K_i = 1$. As it is observed, PI control indicates better response. However, the choice of PI gains were made based on known plant parameters. If the plant parameters were unknown or subject to changes, PI control would not provide satisfactory performance and might even become unstable. When the plant parameters α and β were changed by 20% and 45% respectively, PI/PID control became unstable (Fig. 11). However, Fuzzy control demonstrated a steady performance. Self-tuning was achieved using a Singleton-based control with on-line tuning. Finally, inspection of Fig. 10-12 shows that self-tuning Fuzzy control, in comparison with PI/PID and Fuzzy controls, provides lower steady-state error.

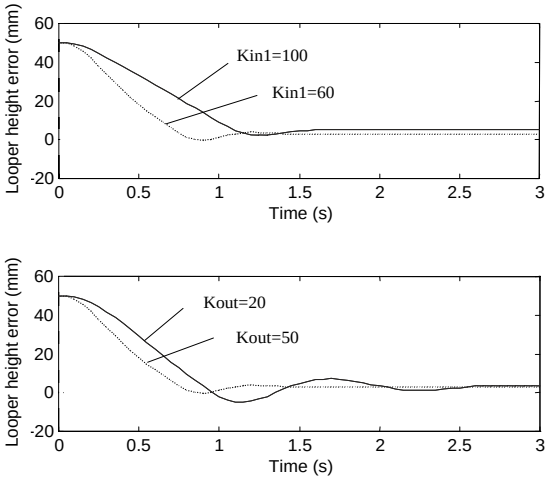

Figure 9: The effect of input/output scaling factors on tuned controller under (C4).

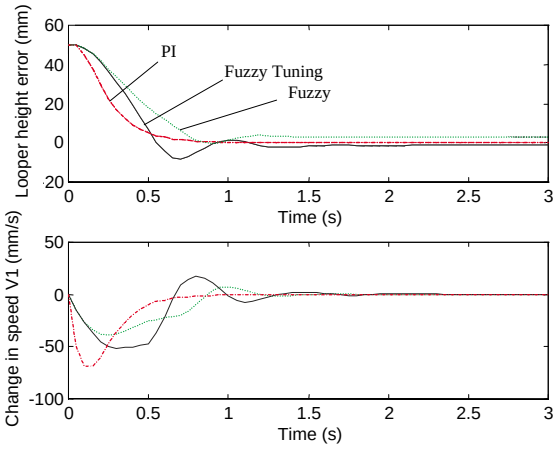

Figure 10: Comparison of the performances of PI, FLC and TFLC under (C3).

6 Conclusions

In this paper, a fuzzy logic controller was proposed for looper control of rolling mills. The proposed system incorporated a learning algorithm for fine-tuning membership functions. Several reasons were given to justify the proposed design. The process behavior is difficult to model, is usually time-varying and depends on parameters which are usually unknown. Therefore, the design of effective classical control laws is difficult and expensive. Alternatively, fuzzy control is relatively simple to design and implement. The simple hardware configuration for fuzzy control is also compatible with most rolling mill installations.

The effectiveness of the proposed system was verified by the simulation results. It was shown that the proposed system is more robust to parameter uncertainties and noises, when compared with classical PI and

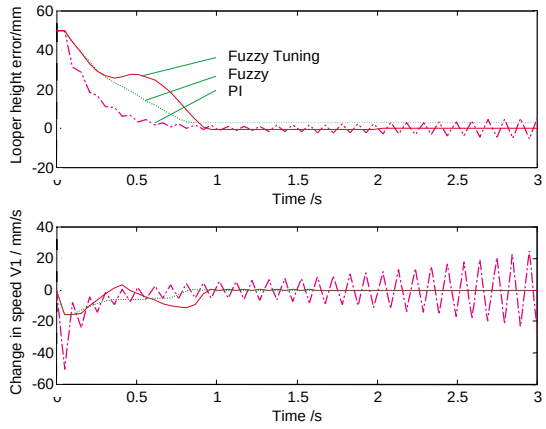

Figure 11: Comparison of the performances of PI, FLC, and TFLC under (C3) with disturbed plant parameters.

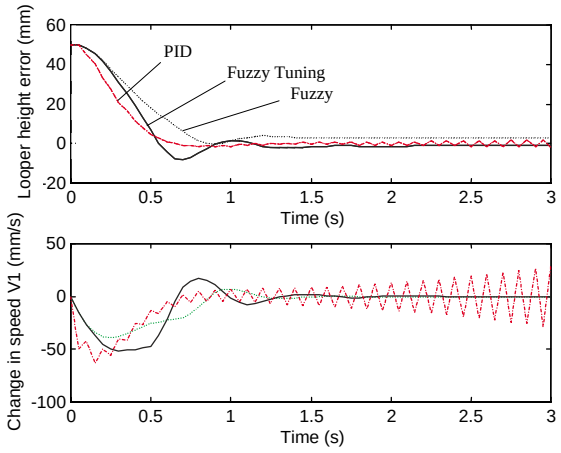

Figure 12: Comparison of the performances of PID, FLC and TFLC under (C3) with disturbed plant parameters.

PID control laws. The incorporated tuning algorithm had the advantage of high-speed learning capability, yet it was effective for improving the control performance. Several practical issues were discussed and conclusions were made regarding different design options. Among them were: off-line vs. on-line tuning, different Jacobian approximations, Mamdani vs. Singleton rules, and various input/output scaling schemes.

Acknowledgments

This research was financially supported by Quad Engineering Inc. Also, we would like to thank the useful discussions and assistance of Leon Winitsky, Richard Li, Jalal Biglou, and Mark Pinkus from Quad Engineering. The second author was also supported by an industrial research fellowship from Natural Sciences and Engineering Research Council of Canada.

References

[1] I. Hayashi, H. Nomura and N. Wakami, "Artificial neural network driven Fuzzy control and its application to the learning of onverted pendulum system," *Proc. 3rd IFSA Congress*, pp. 610-613, 1989.

[2] F. Janabi-Sharifi and J. Fan, "A learning fuzzy system for looper control in rolling mills," *Proc. IEEE Int. Conf. Systems, Man, and Cybernetics: SMC2000*, Nashville, TN, Oct. 2000.

[3] R. Jang, "Self-learning fuzzy controllers based on temporal back propagation," *IEEE Trans. Neural Networks,* vol. 3, pp. 714-723, Sep. 1992.

[4] R. Kruse, J. Gebhardt, and F. Klaonn, *Foundations of Fuzzy Systems.* Wiley, Chichester, 1994.

[5] W. T. Miller, R. S. Sutton, and P. J. Werbos, Eds., *Neural Networks for Control.* Cambridge, MA: The MIT Press, 1990.

[6] K.S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks,* vol. 1, no. 1, pp. 4-27, 1990.

[7] A. Neilson, and E. Koebe, "Microprocessor control system for bar mill revamp at Chapparrel Steel," *Iron and Steel Engineer*, pp. 56-59, Apr. 1983.

[8] H. Nomura, I. Hayashi, N. Wakami, "Self-tuning method of fuzzy reasoning by hopfield neural network," *Proc. 5th Fuzzy System Symp.,* pp. 177-182, 1989.

[9] H. Nomura, I. Hayashi, N. Wakami, "A learning method of fuzzy inference rules by descent method," *Proc. IEEE Int. Conf. on Fuzzy Systems,* pp. 203-210, San Diego, CA, 1992.

[10] T. J. Procyk and E. H. Mamdani, "A linguistic self-organizing process controller," *Automatica*, vol. 15, pp. 15-30, 1979.

[11] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagation errors," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Foundations*, D. E. Rumelhart and J. L. McClelland, Eds., vol. 1, MIT Press, Cambridge, MA, 1986.

[12] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Trans. Systems, Man, and Cybernetics,* vol. 15, no. 1, 1985, pp. 116-132.

[13] L.-X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis.* Englewood Cliffs, NJ: Prentice Hall, 1994.