Soft-Computing Digital Architectures Using von Neumann Stochastic Data Representation

Emil M. Petriu

School of Information Technology and Engineering, University of Ottawa, Ottawa, Ont., K1B Canada

Abstract: This paper reviews the basics of the von Neumann stochastic data representation and its application to the development of digital neural network and fuzzy logic controller architectures.

Keywords: von Neumann, random-pulse machines, stochastic computing, dithering, neural networks, fuzzy logic control.

1 Introduction

While trying to prove that algebraic operations with analog variables can be performed digitally, John von Neumann advanced in 1956 the idea of representing analog variables by the mean rate of random-pulse streams [1].

Random-pulse data appear as sequences of random binary pulses which carry analog information represented by the statistical mean value of the pulse sequence.

Figure 1 shows a basic analog/random-pulse converter. The deterministic analog input V is mixed with an analog dither signal R uniformly distributed between +FS and –FS. The resulting analog random signal VR is then 1–bit quantizied and then sampled by a clock signal CLK to produce the random–pulse sequence VRP. It can be easily shown that the statistical estimation of the deterministic component of this VRP sequence represents a measure of the deterministic analog input V.

As variables are represented by statistical averages of random-pulse streams, the data processing can be done by simple 1-bit arithmetic operations.

Pursuing this idea, a variety of random-pulse processing systems were reported during the last 50 years, [2-9]. On parallel tracks, dither techniques have been used to reduce the effects of the quantization noise in instrumentation and signal processing, [10-16].

An important limitation of the random-pulse is the relatively long time needed to rich an acceptable computational accuracy. However this drawback can be mitigated by increasing the quantization resolution of the dithered signals. The resulting multi-bit data representation will be referred further as the *von Neumann stochastic data representation*.

After a short review of the basic principles of the von Neumann stochastic data representation, the paper presents two soft computing applications of this technique: neural network hardware architecture and fuzzy logic controller architecture developed by the author and his collaborators at the University of Ottawa, [6], [8], [16-18].

2 Von Neumann Stochastic Data Representation

This data representation is a generalization of von Neumann's random pulse representation. A more detailed discussion of this technique and its applications to instrumentation and neural network (NN) architectures can be found in [8].

2.1 Analog / Stochastic Data and Stochastic Data / Digital Conversions

Stochastic data are produced by a multi-bit *analog/stochastic data* converter as shown in Figure 1. Before quantization, the analog signal V, supposed to have a low variation rate relatively to the sampling clock CLK rate, is mixed with an analog dither R uniformly distributed between $+\Delta/2$ and $+\Delta/2$, where Δ is the quantization step, shall fulfil the following statistical requirements: (i) zero mean, (ii) independent of the input V, and (iii) characteristic function having periodic zeros, [10] and [11].

The resulting analog signal VR is quantified with a *b*-bit resolution and then sampled by a clock CLK to produce the stochastic sequence VRD of *b*-bit data with amplitude values between *k-1* and *k*. The ideal estimation over an infinite number of samples of the stochastic data sequence VRD is:

$$E[VRD] = (k-1)\cdot p[(k-1.5)\Delta \leq VR < (k-0.5)\Delta] + k \cdot p[(k-0.5)\Delta \leq VR < (k+0.5)\Delta]$$

$$= (k-1)\cdot\beta + k\cdot(1-\beta) = k - \beta \qquad (1)$$

The estimation accuracy of the recovered value for V depends on the quantization resolution Δ, the finite number of samples that are averaged, and on the statistical properties of the analog dither.

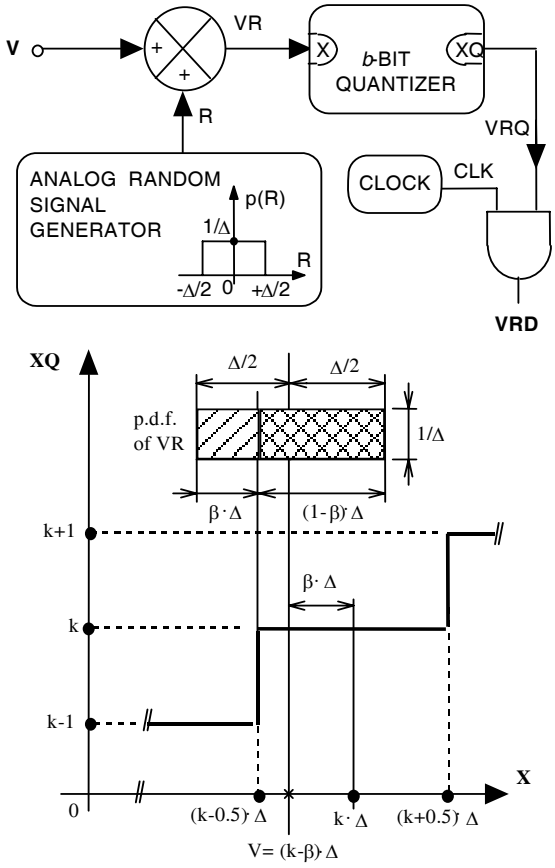


Figure 1
Analog/stochastic data converter

Because of the functional similarity of a neuron and a correlator, we found useful to consider the following table giving relative speed performance figures for correlators with different quantization levels, [10]:

Quantization levels	***Relative mean square error***
2	72.23
3	5.75
4	2.75
...	...
8	1.23
...	...
analog	1

For instance, a 2-level random-pulse correlator will be 72.23 times slower than an ideal analog correlator calculating with the same accuracy the correlation function of two statistically independent Gaussian noise signals with amplitudes restricted within ±3σ. Remarkably, a 3-level (2-bit) correlator will be 5.75 times slower than the analog correlator.

Based on these relative performance figures we are using a 3-level generalized random-data representation produced by a dithered 2-bit dead-zone quantizer, which gives a good compromise between the speed and the circuit complexity, [8].

S*tochastic data / digital* estimates the deterministic component V of a stochastic data sequence by a moving average V^*_N over the most recent *N* samples {VRD_i / i=1,2,...N}:

$$V^*_N = V^*_{N-1} + \frac{VRD_N - VRD_0}{N} \qquad (2)$$

The moving average algorithm eliminates the need to continuously recalculate the sum of the most recent *N* data and then divide it by *N*.

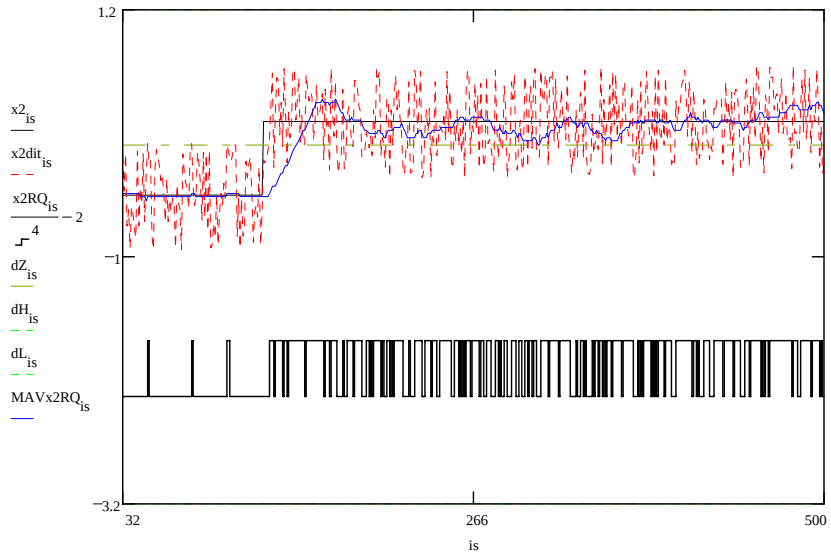


Figure 2

Analog/stochastic data and *stochastic data/digital* conversions (from [8])

Figure 2 shows a step-like analog signal x2 that is converted to a sequence of random data x2RQ, which is then reconverted as a moving average over N=16 samples to produce the analog estimation MAVx2RQ.

2.2 Arithmetic Operations with Stochastic Data

The *arithmetic addition* of m signals $\{x_i \mid i=1,2,...,m\}$represented by their b-bit stochastic data $\{X_i \mid i=1,2,...,m\}$ is be carried out, as shown in Figure 3, by time multiplexing the randomly sampled incoming random-data streams. The uniformly distributed random sampling removes unwanted correlations between sequences with similar patterns, [4]. The stochastic data output sequence $Z = (X_1+...+X_m)/m$ represents the resulting sum signal $z = x_1 +\ldots+ x_m$

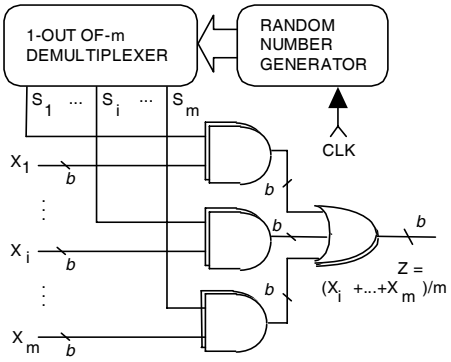


Figure 3
Arithmetic addition of *m* stochastic data streams $X_1, X_2,..., X_m$

The multiplication of two streams of 3-level, 2-bit, unbiased stochastic data is done in parallel by a low complexity combinatorial logic circuit defined by the following equations;

$$Z_{MSB} = X_{LSB} \cdot Y_{MSB} + X_{MSB} \cdot Y_{LSB} \tag{3}$$

$$Z_{LSB} = X_{MSB} \cdot Y_{MSB} + X_{LSB} \cdot Y_{LSB} \tag{4}$$

where X_{MSB} and Y_{MSB} are the most-significant bits, and X_{LSB} and Y_{LSB} are the least-significant bits of the *X* and *Y* random-data samples.

3 Neural Network Architecture Using Stochastic Data Representation

A NN hardware architecture was developed using modular arithmetic operators that process 3-level 2-bit stochastic data streams.

Each synapse multiplies an incoming random data streams X_i, where i=1,2,...,m, by a synaptic-stored weight w_{ij}, which is adjusted during the learning phase.

The neuron-body integrates the $DT_{ij} = X_i * w_{ij}$ signals from all the incoming post-synaptic channels, as illustrated in Figure 4. In order to apply the activation function, the results of this integration are converted to a digital representation. A final digital/stochastic data conversion is then used to restore the stochastic data format for the output Y_j of the neuron.

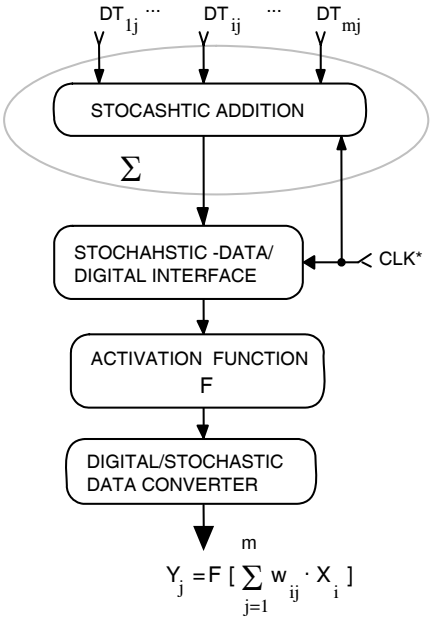


Figure 4
Neuron body structure using stochastic data representation

Using this stochastic data neuron, we have implemented a 30 input auto-associative memory NN for pattern recognition applications, shown in Figure 5, [17]. This 30 input auto-associative memory NN is able to recognize any of the initially taught associations. If it receives an input $P=P_q$ then it produces an output $a = t_q$, for $q = 1,2,...,Q$.

The training set consists of three training patterns, which represent the digits {0,1,2} displayed as a 6x5 grid. Each white square is represented by a '-1', and each black square is represented by a '1'. The weight matrix in this case is $W = P_1P_1^T + P_2P_2^T + P_3P_3^T$.

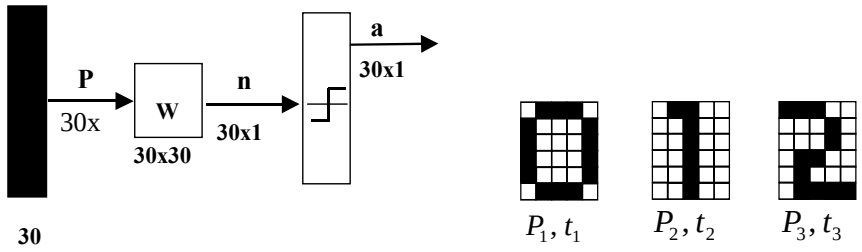


Figure 5
The auto-associative memory NN and the training set (from [8])

The auto-associative memory NN can also recognize patterns corrupted by noise: i.e. if the input is changed $P = P_q + \delta$ the output will still be $a = t_q$. Tests have shown that it is able to deal with up to 30% noise-corrupted patterns as illustrated in Figure 6.

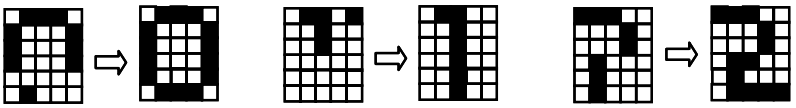


Figure 6
Recovery of 30% occluded patterns by the auto-associative NN (from [8])

4 Fuzzy Logic Controller Architecture Using Stochastic Data Representation

While there are only few fuzzy membership functions covering the input and output domains of a *fuzzy logic controller* (FLC), the overlapping of the fuzzy domains and their linear membership functions will eventually allow the achievement of the desired high-resolution I/O function between crisp input and output variables.

The low-bit stochastic data representation allows us to implement digital FLC architectures which use fewer logic circuits than the traditional digital architectures providing the same high-resolution I/O function.

4.1 FLC for Backing-up a Four Wheel Truck

The problem, illustrated in Figure 7, is to back up a truck into a docking station from any initial position that has enough clearance from the docking station, [16].

The 35 rule Sugeno-style FLC, Figure 8, has two input variables: the truck angle φ and the x position. The output variable is the steering angle θ.

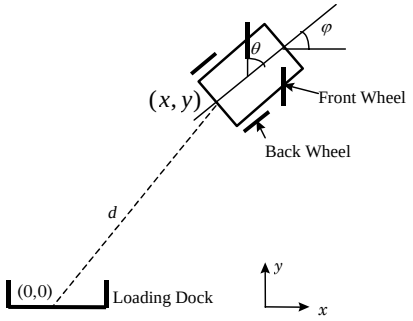
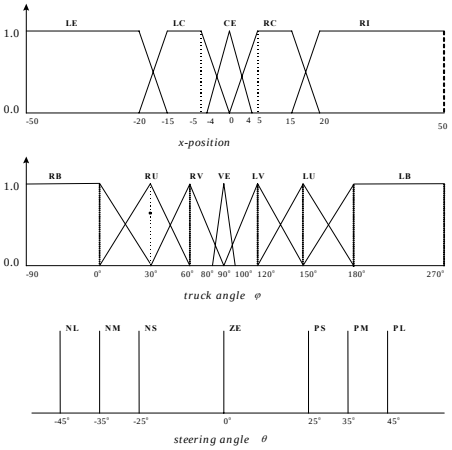


Figure 7
The parameters of the truck backing-up problem



φ \ x	LE	LC	CE	RC	RI
RL	NL [1]	NL [2]	NM [3]	NM [4]	NS [5]
RU	NL [6]	NL [7]	NM	NS	PS
RV	NL	NM	NS	PS	PM
VE	NM	NM	ZE [18]	PM	PM
LV	NM	NS	PS	PM	PL
LU	NS	PS	PM	PL	PL [30]
LL	PS [31]	PM [32]	PM [33]	PL [34]	PL [35]

Figure 8
The membership functions and the rule-base for the FLC (from [16])

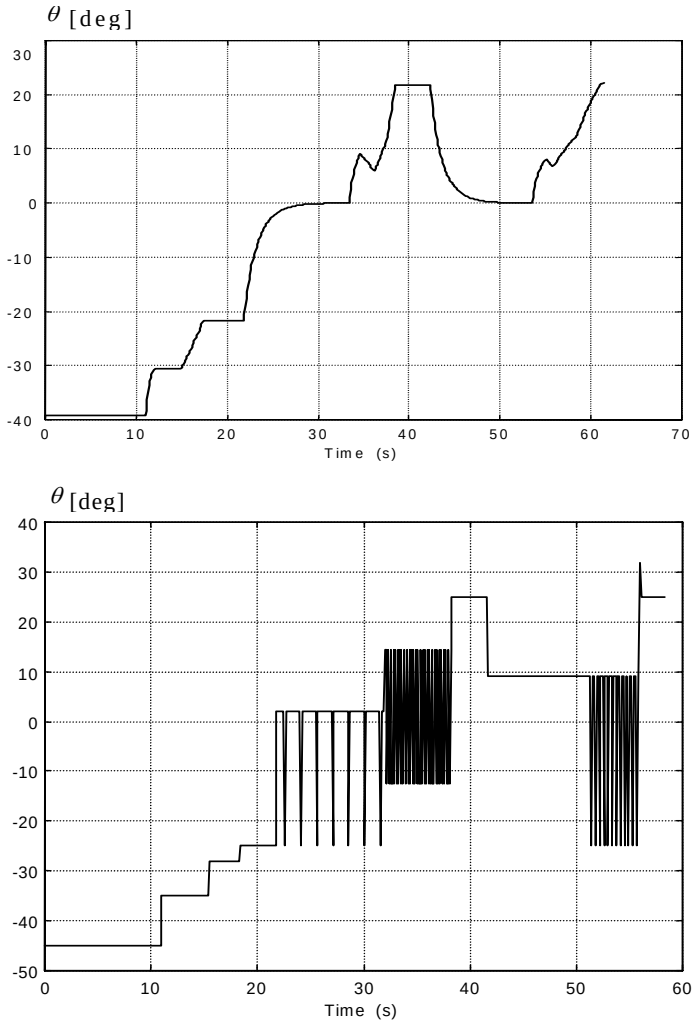


Figure 9
The FLC output θ during docking when the input variables, φ and x are analog and respectively digitally quantizied with a 4-bit bit resolution (from [16])

As illustrated in Figure 9, the steering angle θ delivered at the output of a 4-bit purely digital FLC is too jerky for any practical application.

The quality of the FLC output exhibits a remarkable improvement if we use a stochastic data representation for the same 4-bit internal resolution, Figure 10.

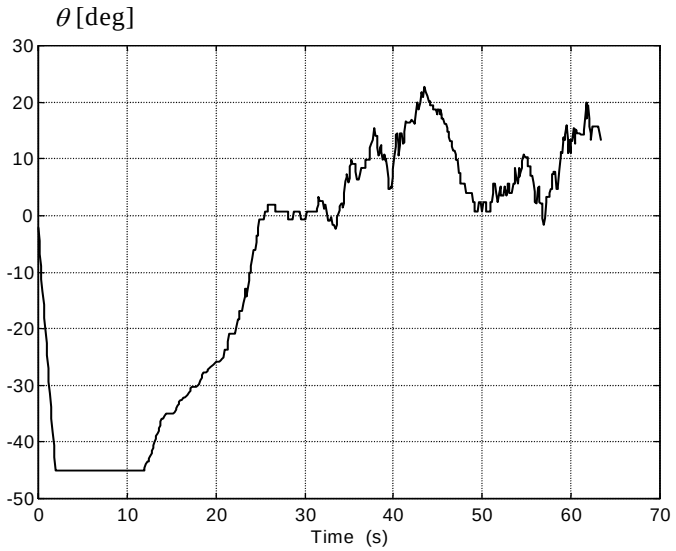


Figure 10
The FLC output θ during docking when the input variables, φ and x, are represented by 4-bit stochastic data (from [16])

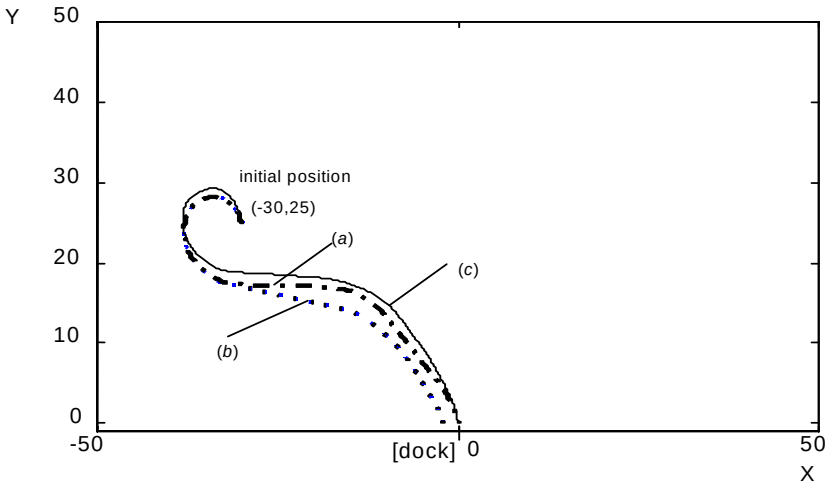


Figure 11
Truck trails for different FLC architectures: (a) analog ; (b) 4-bit purely digital, and (c) 4-bit stochastic data with 20-unit moving average filter (from [16])

Figure 11 shows comparatively the trails of a truck docking under the control of (*a*) an analog FLC, (*b*) a purely digital 4-bit FLC, and (*c*) a FLC using 4-bit

stochastic data and a 20-unit moving average low-pass filter at the controller's output. It may be interesting to note that while both trails (*a*) and (*c*) end as requested at the loading dock (0,0), the trail (*b*) does not reach the dock in the end.

Conclusions

Due to its relatively low hardware complexity and high internal noise immunity, the stochastic data processing represents an attractive alternative to the analog techniques for many statistical signal processing and soft computing applications.

In order to decide on the optimal number of bits for the stochastic data representation one needs to solve a classical time-versus-complexity tradeoff depending on the specific intended application, as well as on the type and cost of the employed technology.

Acknowledgement

Using the von Neumann stochastic data representation as a unifying perspective, this paper represents a synthesis of the work carried out by the author and his collaborators at the University of Ottawa and published over the years in conference proceedings and journals: [6], [8], [16-18].

The work reported in this paper was funded in part by Communications and Information Technology Ontario (CITO) and the Natural Sciences and Engineering Research Council (NSERC) of Canada.

References

[1] J. von Neumann: Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components, in Automata Studies, (C.E. Shannon, Ed.), Princeton University Press, Princeton, NJ, 1956, pp. 43-98

[2] W. J. Poppelbaum, C. Afuso: Noise-Computer, University of Illinois, Urbana, Dept. Comp. Sc., Quart. Tech. Progress Reports, Jan.-March 1965, April-June 1965, July-Sept. 1965, Jan.-March 1966

[3] S. T. Ribeiro: Random-Pulse Machines, in IEEE Trans. Electron. Comp., Vol. EC-16, No. 3, June 1967, pp. 261-276

[4] B. R. Gaines: Stochastic Computer Thrives on Noise, in Electronics, July 1967, pp. 72-79

[5] A. Hamilton, A. F. Murray, D. J. Baxter, S. Churcher, H. M. Reekie, L. Tarasenko: Integrated Pulse Stream Neural Networks: Results, Issues, and Pointers, in IEEE Trans. Neural Networks, Vol. 3, No. 3, May 1992, pp. 385-393

[6] E. Petriu, K. Watanabe, T. Yeap: Applications of Random-Pulse Machine Concept to Neural Network Design, in IEEE Trans. Instrum. Meas., Vol. 45, No. 2, 1996, pp. 665-669

[7] S. Naess, T. S. Lande: Building Blocks for Low-Power Stochastic Pulse Coded Systems, in Proc. ESSCIRC'07 - The 23rd European Solid State Circuits Conf., 1997, pp. 164-167

[8] E. M. Petriu, L. Zhao, S. R. Das, V. Z. Groza, A. Cornell: Instrumentation Applications of Multibit Random-Data Representation, in IEEE Trans. Instrum. Meas., Vol. 52, No. 1, 2003, pp. 175- 181

[9] S. Sato, K. Nemoto, S. Akimoto, M. Kinjo, K. Nakajima: Implementation of a New Neurochip Using Stochastic Logic, in IEEE Trans. Neural Networks, Vol. 14, No. 5, Sept. 2003, pp. 1122-1127

[10] K-.Y. Chang, D. Moore: Modified Digital Correlator and its Estimation Errors, in IEEE Trans. Inf. Theory, Vol. IT-16, No. 6, 1970, pp. 699-706

[11] F. Castanie: Signal Processing by Random Reference Quantizing, in Signal Processing, North Holland, Vol. 1, No. 1, 1979, pp. 27-43

[12] E. Pop, E. Petriu: Influence of Reference Domain Instability Upon the Precision of Random Reference Quantizer with Uniformly Distributed Auxiliary Source, in Signal Processing, North Holland, Vol. 5, 1983, pp. 87-96

[13] W. Chou, R. M. Gray: Dithering and its Effects on Sigma-Delta and Multistage Sigma-Delta Modulation, in IEEE Trans. Inf. Theory, Vol. 37, No. 3, 1991, pp. 500-513

[14] R. M. Gray, T. G. Stockham: Dithered Quantizers, in IEEE Trans. Inf. Theory, Vol. 39, No. 3, 1993, pp. 805-912

[15] P. Carbone, D. Petri: Performance of Stochastic and Deterministic Dithered Quantizers, in IEEE Trans. Instrum. Meas., Vol. 49, No. 2, 2000, pp. 337-340

[16] E. M. Petriu, J. Mao: Fuzzy Sensing and Control for a Truck, in Proc. VIMS-2000, IEEE Workshop on Virtual and Intelligent Measurement Systems, Annapolis, MD, April 2000, pp. 27-32

[17] L. Zhao: Random Pulse Artificial Neural Network Architecture, M.A.Sc. Thesis, University of Ottawa, Canada, 1998

[18] J. Mao: Reduction of the Quantization Error in Fuzzy Logic Controllers by Dithering, M.A.Sc. Thesis, University of Ottawa, Canada, 1998