Neural Network Linearization of Pressure Force Sensor Transfer Characteristic

Jozef Vojtko, Irena Kováčová, Ladislav Madarász, Dobroslav Kováč

Faculty of Electrical Engineering and Informatics, Technical University of Košice, Letná 9, 042 00 Košice, Slovak Republic
Jozef.Vojtko@tuke.sk, Irena.Kovacova@tuke.sk, Ladislav.Madarasz@tuke.sk, Dobroslav.Kovac@tuke.sk

Abstract: The paper deals with an elastomagnetic sensor of pressure force and neural network design in order to achieve linear sensor output. There are described basic properties of such sensor and its equivalent electrical scheme. The feeding and evaluating circuits were designed in order to obtain the optimal working conditions.

Keywords: measurement, elastomagnetic sensor, neural network, non-linearity, hysteresis

1 Introduction

Elastomagnetic sensors have become more widespread owing to their extensive use in industrial and civil automation. However, designing low-cost and accurate sensors still requires great theoretical and experimental efforts to materials engineers. But this task can be solved by advanced electronic techniques for automatic calibration, linearization and error compensation.

2 Basic Properties of Elastomagnetic Sensor

The elastomagnetic sensor [EMS] of a pressure force that utilizes the Villary´s phenomena principle, which consists of the fact that if a ferromagnetic body is subjected to mechanical stress, its form is changed and consequently its permeability is changed, too [1]. Villary´s principle is based on equation:

$$\left(\frac{\partial M}{\partial p}\right)_{H,\vartheta} = \left(\frac{\partial w}{\partial H}\right)_{p,\vartheta} \tag{1}$$

where M is magnetic polarization, p is general pressure, w is relative deformation, H is intensity of magnetic field, ϑ is ambient temperature.

We can state magnetostriction coefficient in saturation for cubic crystal shown in Fig. 1 by following equation:

$$\lambda_s = h_1\left(\sum_{i=1}^{3}\alpha_i^2\beta_i^2 - \frac{1}{3}\right) + 2h_2\left(\alpha_1\alpha_2\beta_1\beta_2 + \alpha_1\alpha_3\beta_1\beta_3 + \alpha_3\alpha_2\beta_3\beta_2\right) +$$

$$+ h_4\left(\sum_{i=1}^{3}\alpha_i^4\beta_i^2 + \frac{2}{3}A - \frac{1}{3}\right) + 2h_5\left(\alpha_1\alpha_2\alpha_3^2\beta_1\beta_2 + \alpha_2\alpha_3\alpha_1^2\beta_2\beta_3 + \alpha_1\alpha_3\alpha_2^2\beta_3\beta_1\right) +$$

$$+ h_3\left(A - \frac{1}{3}\right) \quad \text{if: } K_1 < 0 \quad \text{for example Ni} \qquad (2)$$

$$+h_3 A \quad \text{if: } K_1 > 0 \quad \text{for example Fe}$$

where $A = \alpha_1^2\alpha_2^2 + \alpha_2^2\alpha_3^2 + \alpha_3^2\alpha_1^2$, K_1 is first anisotropic constant, $(\alpha_1, \alpha_2, \alpha_3), (\beta_1, \beta_2, \beta_3)$ are cosine functions of angles created by vectors of magnetic field and magnetostriction in saturation state, $h_1 - h_5$ are magnetostriction parameters which were stated experimentally. According to Fig. 1, we can state $\alpha_i = \cos\varphi_i$ and $\beta_i = \cos\gamma_i$. Also we are able to simplify the above mentioned equation by the fact that parameters h_1 and h_2 have few times greater values than rest parameters so the rest parameters can be negligible.

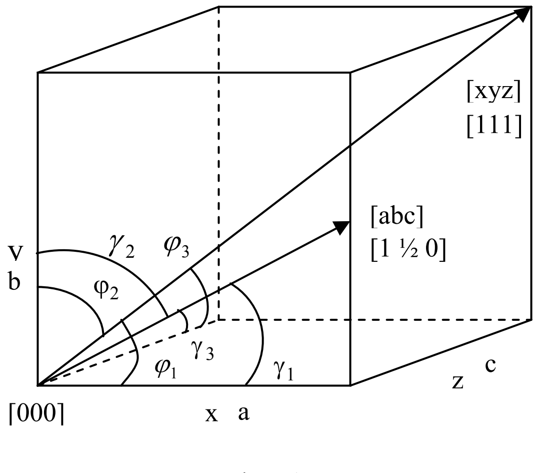


Figure 1
Single cubic crystal

The resulting magnetostriction in directions <100> and <111> will be given as:

$$\lambda_s = \frac{3}{2}\lambda_{100}\left(\sum_{i=1}^{3}\alpha_i^2\beta_i^2 - \frac{1}{3}\right) + 3\lambda_{111}\left(\alpha_1\alpha_2\beta_1\beta_2 + \alpha_1\alpha_3\beta_1\beta_3 + \alpha_3\alpha_2\beta_3\beta_2\right) \tag{3}$$

Next relation describes dependency between magnetic induction and magnetic intensity:

$$B = (\mu + \Delta\mu)H \tag{4}$$

where $\Delta\mu$ represents the increment of permeability caused by acting of external pressure force. The next relation can be obtained by comparing of increments of magnetic and elastomagnetic energies:

$$\frac{1}{2}\Delta\mu H^2 = \sigma\lambda_m \tag{5}$$

where λ_m represents a magnetostriction coefficient. It is defined like:

$$\lambda_m = \frac{3}{2}\lambda_s \frac{M^2}{M_s^2} \tag{6}$$

It generally holds that:

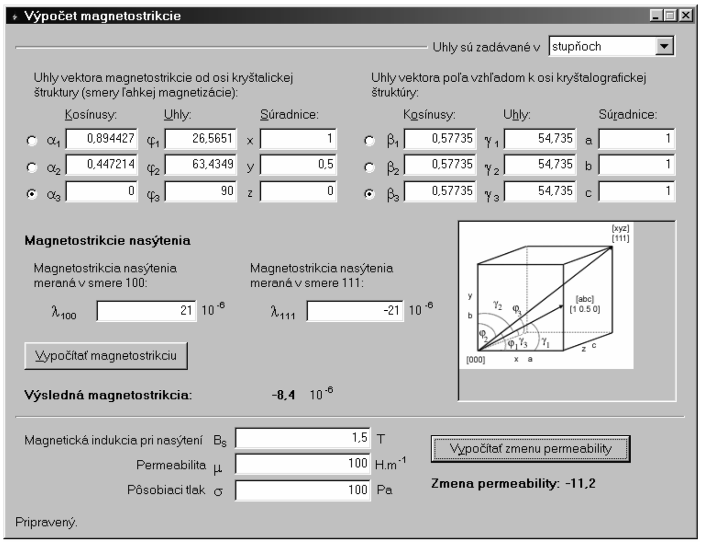
$$\frac{M}{M_s} = \frac{B}{B_s} \tag{7}$$

By utilizing of the last two equations we can obtain the final dependence for permeability increment caused by the acting of external pressure force [2]:

$$\Delta\mu = 3\frac{\sigma\lambda_s\mu^2}{B_s^2} \tag{8}$$

For permeability increment calculation we can utilize software calculator shown in Fig. 2. Since the permeability determines the magnetic field in a ferromagnetic body, so the magnetic field is also changed and we could measure its changes by changes of the induced electric voltage.

On the base of the above mentioned one can see that the pressductor can be described as a transformer in which the mutual inductance between the primary and secondary windings is changed proportionally to the acting stress or to the pressure, but only in the case if magnetizing current I_m is constant and output current I_s is negligible.



Výpočet magnetostrikcie

Uhly sú zadávané v: stupňoch

Uhly vektora magnetostrikcie od osi kryštalickej štruktury (smery ľahkej magnetizácie):

	Kosínusy:		Uhly:		Súradnice:
○ α_1	0,894427	φ_1	26,5651	x	1
○ α_2	0,447214	φ_2	63,4349	y	0,5
● α_3	0	φ_3	90	z	0

Uhly vektora poľa vzhľadom k osi kryštalografickej štruktúry:

	Kosínusy:		Uhly:		Súradnice:
○ β_1	0,57735	γ_1	54,735	a	1
○ β_2	0,57735	γ_2	54,735	b	1
● β_3	0,57735	γ_3	54,735	c	1

Magnetostrikcie nasýtenia

Magnetostrikcia nasýtenia meraná v smere 100: λ_{100} 21 10^{-6}

Magnetostrikcia nasýtenia meraná v smere 111: λ_{111} -21 10^{-6}

Výsledná magnetostrikcia: **-8,4** 10^{-6}

Magnetická indukcia pri nasýtení B_S: 1,5 T

Permeabilita μ: 100 $H.m^{-1}$

Pôsobiaci tlak σ: 100 Pa

Zmena permeability: **-11,2**

Pripravený.

Vypočítať magnetostrikciu
Vypočítať zmenu permeability
[xyz]
[111]
[abc]
[1 0.5 0]
[000]
x a
y b
z c
γ_1 γ_2 γ_3
φ_1 φ_2 φ_3

Figure 2
Intelligent calculator

The elastomagnetic sensor equivalent electrical scheme is shown in Fig. 3.

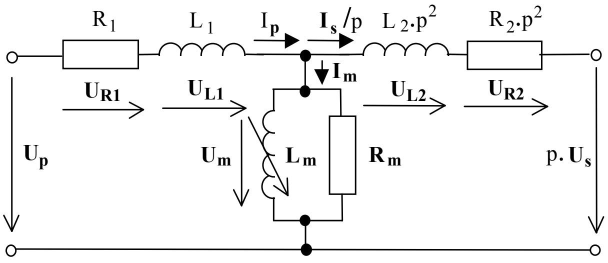


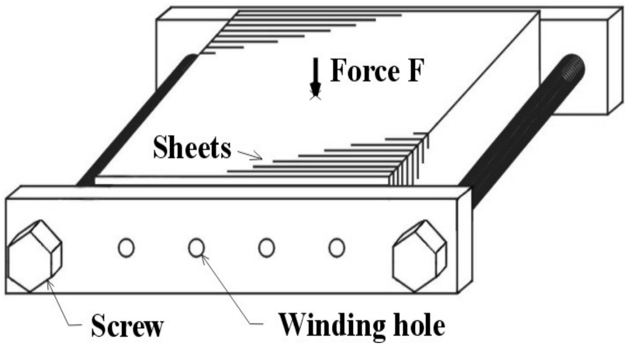
Figure 3
Elastomagnetic sensor equivalent electrical scheme

The change of the output voltage value can be calculated by following equation:

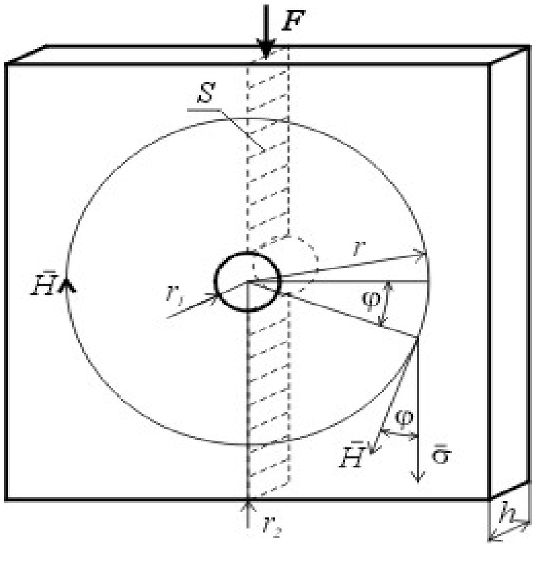
$$\Delta U_s = 4 \cdot 2\pi \cdot f \cdot N_s \cdot \frac{N_p I_p (r_2 - r_1) h}{2\pi \dfrac{r_2 - r_1}{\ln \dfrac{r_2}{r_1}}} \cdot \frac{2}{\pi} \cdot \frac{2\lambda_s \mu^2}{B_s^2} \cdot$$

$$\cdot \frac{1}{2 r_2 h} \cdot F = \frac{8 f N_s N_p I_p \lambda_s \mu^2}{B_s^2 \pi r_2} \cdot \ln \frac{r_2}{r_1} \cdot F \tag{9}$$

This equation corresponds with practical manufacturing of elastomagnetic sensor in Fig. 4 [3]. Properties of the ferromagnetic material have a significant influence on sensor sensitivity, mainly: saturation magnetostriction coefficient λ_s, permeability of material μ and saturation intrinsic induction B_s (for elastomagnetic sensor EMS-200kN: $\lambda_s = 2.15 \cdot 10^{-6}$, $\mu = 9.55 \cdot 10^{-4}$ H/m, $B_s = 1.28$ T).



a)



b)

Figure 4
Sketches of elastomagnetic sensor
a) composed sensor EMS-200kN, b) detail of sheet element

The described circuit fully corresponds to the transformer. In this case, the relation between magnetic intensity and magnetic induction is given by nonlinear hysteresis curve.

The maximum useful signal is obtained if output current I_s is reduced to the minimum and if the influence of primary current I_p effective value is eliminated. In this case, the magnetizing voltage U_m corresponds to the maximum output voltage U_s for given operating point which is depending on the primary current I_p value and the acting force. Such a way can be reduced the power of the feeding source.

3 A Design of the Feeding and Evaluating Circuits

The feeding circuit must fulfill basic condition which consists in the current feeding request, because only in this case, the change of acting pressure force on the elastomagnetic sensor core will be represented by the change of output secondary voltage U_s [4]. An example of such feeding source realization is shown in Fig. 5. Such a way is simply possible to secure realization of the harmonic constant current source by step down line voltage transformer with small output power.

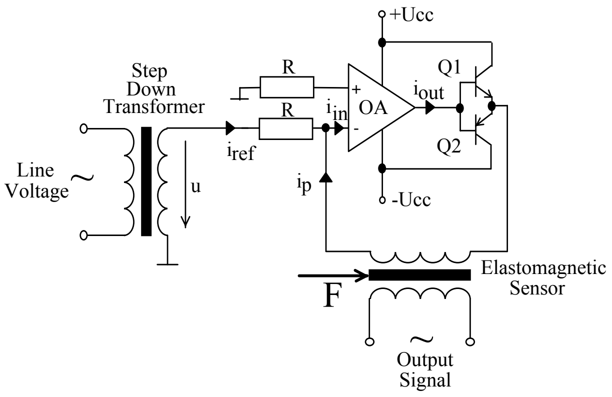


Figure 5
An example of the optimal feeding source

For second request fulfilling, which is concerning to the secondary winding current I_s minimum value we must to secure as high as possible input impedance of evaluating circuit. A simple and suitable output evaluating subcircuit can be realized by OA as it is shown in Fig. 6.

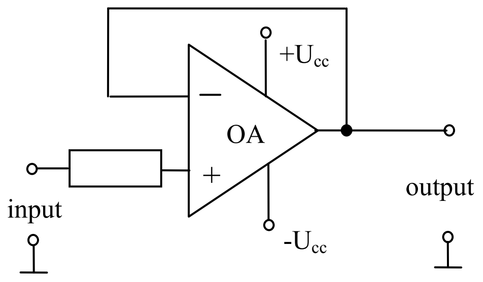


Figure 6
An example of output subcircuit connection with high impedance

4 Neural Network Design

The neural network (NN) is expected to eliminate transformer nonlinearity. However, NN output should be linear and expressed by equation of straight line. In order to achieve this aim, several NN models were designed. The differences between linear output and the real sensor output are shown in the Fig. 7. The characteristics ΔU_i is gained from output sensor voltage $U_{2\uparrow} = f(F)$ (if force F was increasing from 0 kN to 200 kN – characteristic upward) and characteristics ΔU_d is gained from $U_{2\downarrow} = f(F)$ (if force F was decreasing from 200 kN to 0 kN – characteristic downward). It can be expressed by next equation:

$$\Delta U_i = U_{2\uparrow} - U_{lin} \tag{10}$$

$$\Delta U_d = U_{2\downarrow} - U_{lin} \tag{11}$$

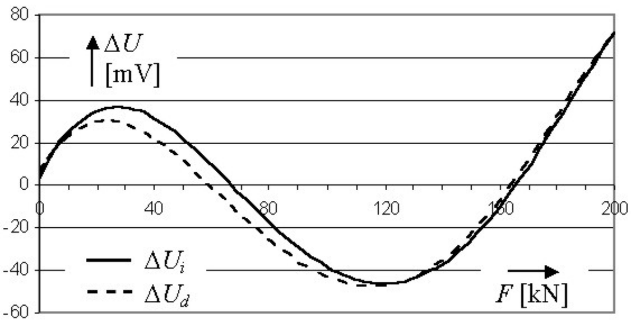


Figure 7
Differences between the linear and the real sensor outputs

The NN task is to reduce the deviation between $U_2\uparrow$, $U_2\downarrow$ and linear regression of sensor output U_{lin}. Finally, the differences ΔU_i and ΔU_d will be limited. The most common artificial neural network, called multilayer feed-forward neural network (FFNN) was used for this purpose [5]. Conception of FFNN with one-unit time delay is shown in Fig. 8.

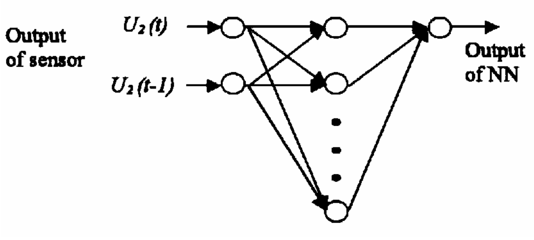


Figure 8
The conception of NN

The sensor output is at the same time the NN input. However, in this proposal the two NN input neurons are used. The both are directly connected to sensor via ADC converter, but the second one is time delayed. A decreasing of sensor errors is expected by using this NN connection.

5 Training Process

The topology of NN consists of 10 neurons in hidden layer, which seems to be the most convenient according to computing speed and accuracy. There were 20 000 training cycles used. Like a learning algorithm the backpropagation was used and it offers an effective approach to the computation of the gradients [6], in Fig. 9. The learning parameter α, which specifies the step width of the gradient descent, was changed in the wide range (see Fig. 10). Here is the SSE (sum of square errors) dependence on training cycles. As we can see, the training process with higher learning parameter achieves smaller SSE at the constant number of training cycles.

If α parameter was more than 1, the sum square error (SSE) of training set was decreased rapidly (the NN respond to trained data was good), but SSE error of test set was increased (the NN respond to untrained data was bad) – over-trained NN.

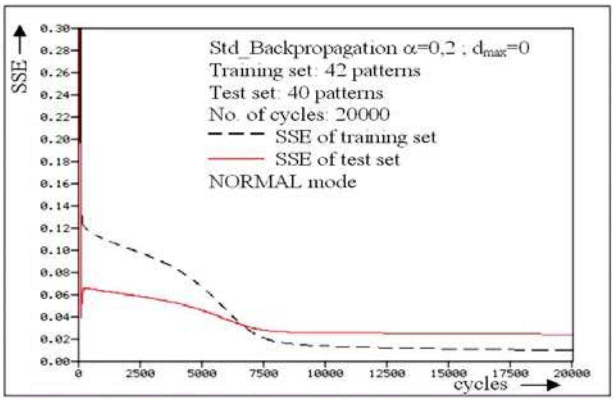


Figure 9

The training process

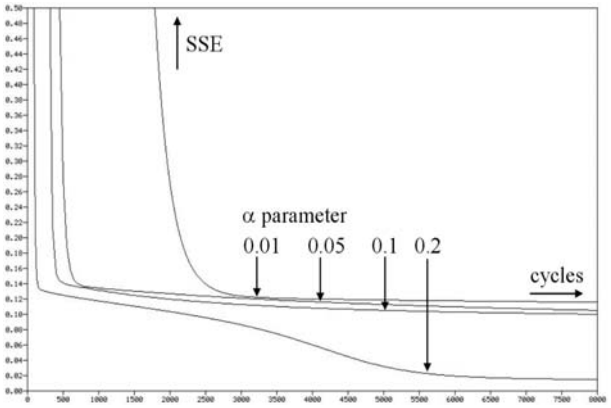


Figure 10

The training process with different α parameter

The output of NN was oscillated at larger α parameter, so the stability of NN was not guaranteed. Advantages of higher NN learning rate was the decreasing of training cycles and at the same time the decreasing of SSE error of training set, but big disadvantages were: over-trained NN, bad generalization, oscillations of NN and instability of NN.

Conclusions

Such construction of elastomagnetic pressure force sensor is predetermined for hard field conditions and aggressive corroding media. Its output signal is even 1000 times greater as signal of resistance transducers and this fact enables to simplify feeding and data evaluation. Such sensors are also less sensitive against extremist electromagnetic interferences. A general construction of these sensors can be realized with smaller costs and dimensions.

The neural network simulator SNNS v4.1 was used for simulation of designed NN. The NN should have decreased the sensor error and its output should have been a linear function. Fig. 11 shows the difference between tested data U_{test} and linear regression U_{lin} ($\Delta U_{test} = U_{test} - U_{lin}$), and difference between NN model data U_{NN} and U_{lin} ($\Delta U_{NN} = U_{NN} - U_{lin}$).

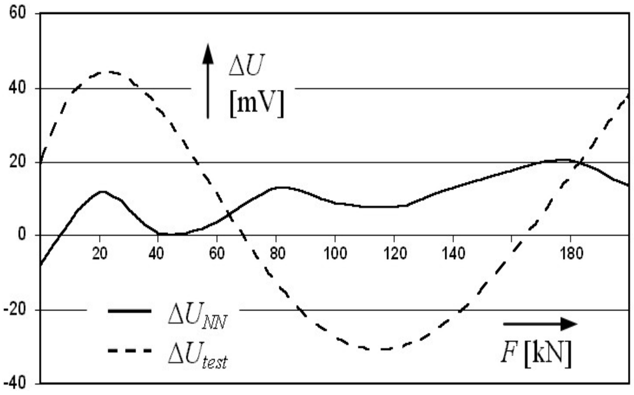


Figure 11
Differences between tested data, NN output and linear regression

The nonlinearity of sensor output was δ_S = 4,34% (for tested data δ_S = 2,69%). The nonlinearity of designed model was δ_{NN} = 1,25% in comparison with a classical FFNN model (without one-unit time delay) where the nonlinearity was δ_{NN} = 1,53%. The finally, the designed model of error correction of elastomagnetic sensor by using FFNN (with one-unit time delay) achieves quantitatively lower linearity error in comparison with real sensor output.

Acknowledgement

The authors gratefully acknowledge the contributions of Slovak Grant Agency as project No.1/0376/2003 and Institutional project No. 4433 of Faculty of Electrical Engineering and Informatics, Technical University of Košice.

References

[1] M. Mojžiš et al.: Properties of 200 kN Force Sensor. *Journal of Electrical Engineering*, Vol. 50, No. 3-4, 1999 pp. 105-108

[2] M. Peťko: Obtainment of Prime Magnetisa-tion Work Values and Magnetisation Work Values by Using Approximate Functions. In *Proceedings of the II Doctoral conference*, TU FEI Košice, 2002, pp. 59-60

[3] M. Mojžiš, M. Orendáč, J. Vojtko: Pressure Force Sensor. In *Proceedings of the II Internal scientific conference*, TU FEI Košice, 2001, pp. 19-20

[4] D. Kováč: Feeding and Evaluating Circuits for an Elastomagnetic Sensor. *Journal of Electrical Engineering*, Vol. 50, No. 7-8, 1999, pp. 255-256

[5] S. Haykin: Neural Networks (A Comprehensive Foundation). *Macmillan College Publishing Company Inc.*, ISBN 0-02-352761-7, 1994

[6] M. Kuczmann, A. Ivanyi: A new neural-network-based scalar hysteresis model. *IEEE Transactions on Magnetics*, Vol. 38, 2002, pp. 857-860

[7] V. Kvasnička, Ľ. Beňušková, J. Pospíchal, I. Farkaš, P. Tiňo, V. Kráľ: Introduction to neural networks theory (Úvod do teórie neurónových sietí, in Slovak), *Iris Publisher*, Bratislava, 1997

[8] A. Zell et al.: SNNS User Manual, version 4.2. *University of Stuttgart, Institute for Parallel and Distributed High Performance Systems; University of Tübingen*, Since 1989

Torque Ripple Calculation of the Two-phase Permanent Magnet Synchronous Motor Supplied by a Triac Converter

Pavel Záskalický

Department of Electrical Drives and Mechatronics, Faculty of Electrical Engineering and Informatics, e-mail: pavel.zaskalicky@tuke.sk

Mária Záskalická

Department of Applied Mathematics, Faculty of Mechanical Engineering, e-mail: maria.zaskalicka@tuke.sk

Technical University of Košice, Letná 9, 04120 Košice, Slovakia

Abstract: A synchronous motor with ferrite permanent magnet rotor is a good solution for small pump applications. It also has some drawbacks. The most important of them seems to be its inability to start directly on the mains, Permanent magnet motor has to by equipped with an electronic circuit for direct starting, which increases motor price. Another drawback is the torque ripple for the non-harmonic supply. This paper shows analytical calculation of the torque ripple of the small permanent magnet motor, which does a triac converter supply. The converter forms two-phase supply voltage of 25 Hz frequency, from one-phase 50Hz mains.

Keyword: torque ripple, synchronous motor, permanent magnet, triac converter

1 Introduction

Small water pump having rated power about 100W have a large application in automobile industry, central heating and house appliances (washing machines, dishwasher). They are run either by single-phase ac motor, when operated from the mains or by a dc motor in the case in an automobile.

Both, motor and pump are often manufactured as a single piece of equipment. A separated water pump in appliances tends to leak water between the rotating shaft and casing. In an integral motor pump a stainless-steel cylinder has to be inserted into the air-gap to protect the stator from water, and the whole rotor body is inserted into another stainless steel cylinder, to protect the rotor cage and lamination. The stator and rotor cylinders cause additional losses and significantly deteriorate motor performance.

Water leakage problems and low efficiency of an induction motor are the main reasons why another concept of the water pump was developed, in which the rotor of the motor is immersed into pumping water. In this solution the rotor of the motor is exposed to chemically aggressive water. A squirrel-cage rotor cannot be used, but a ferrite permanent magnet rotor seems to be a good choice. Strontium-ferrite permanent magnets are chemically inert, which make them suitable for applications in aggressive environments. Strontium-ferrite magnets have high specific electric resistance, so they do not experience thermal problems due to eddy-current losses. Their low residual flux density imposes the need for special machine construction when high air-gap flux density is needed.

2 Motor Configuration

Figure 1 shows the sketch of the construction disposition of a two-phase synchronous motor having permanent magnets on the rotor. The stator structure is similar to that of a two-phase salient poles reluctance motor. The stator magnetic circuit is built from the laminations. The rotor form the two-pole cylinder permanent magnet. The Stroncium-Ferrite magnet with linear demagnetisation characteristic and remanence 0.5 T was used.

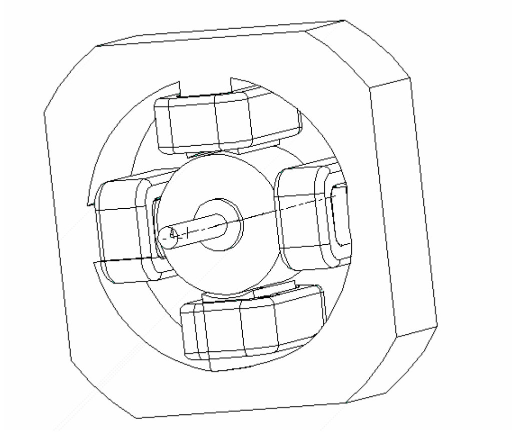


Figure 1
Two-phase permanent magnet motor

The stator windings are of particularly simple form. Two opposite placed winding form one phase. The stator windings can be configured to either a serial or parallel two-phase system. Normally the windings are identical. The windings which forms one phase of the motor are connected like shown on Fig. 1. Two diametrically opposite stator poles are of opposite magnetic polarity.

The electromagnetic torque is mainly developed due to the interaction between stator winding current and rotor permanent magnet flux.

Electromagnetic torque can by calculated with the two corresponding temporal quantities that are phase current and back electromotive force (emf), or with the related spatial quantities that are stator magnetomotive force (mmf) and the rotor induction.

Finite-element analysis was employed for a basic design approach predicting static performance of the motor. The wave of induced emf in stator coil windings was numerically calculated from the wave of the flux. The calculation results that the waveform of the emf can by mathematically substituted by a sinusoidal function.

3 Mathematical Model

The analytical description of two-phase synchronous motor with permanent magnet rotor is simpler than that of tree-phase, due to the fact that stator windings of such two-phase motors are physically orthogonal and thus magnetically decoupled. The mathematical description bears strong resemblance to its single-phase counterpart.

Figure 2 shows the per-phase equivalent circuit of the machine.

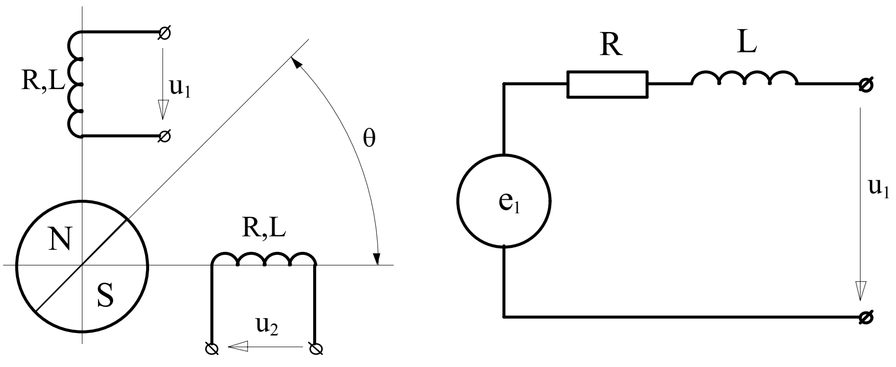


Figure 2
Equivalent circuit

Let us assume that the reluctance torque is negligible. This one depends on the air-gap between the poles. In accordance with the stator cross section shown in the Figure 1, the reluctance torque would exhibit four-maximums and four-minimums per complete rotation.

Assuming that all windings are identical and the magnetic circuit is symmetrical.

Instantaneous value of electrical input power is determined as [1],[2],[3]:

$$p = u_1 i_1 + u_2 i \tag{1}$$

It consists of three parts:

$$p = p_j + p_m + p_e \tag{2}$$

$p_j = R\left(i_1^2 + i_2^2\right)$ represents the copper losses in the stator coil;

$p_m = L\left(i_1 \frac{\mathrm{d}i_1}{\mathrm{d}t} + i_2 \frac{\mathrm{d}i_2}{\mathrm{d}t}\right)$ represents the magnetic reactive power;

$p_e = e_1 i_1 + e_2 i_2$ represents the electrical output-power.

Where: R - is the armature resistance; L - is the synchronous inductance; e_1, e_2 - phase electromotive forces;

The product of the torque and the speed gives the electrical output power of the machine:

$$p_e = m.\omega \tag{3}$$

Where: m - instantaneous value of the torque, ω- speed of the machine.

To determine the waveform of the torque of the machine, it's necessary to determine the waveform of the phase currents. These one can by calculated from the voltage equations [4]

$$\begin{aligned} u_1 &= Ri_1 + L\frac{\mathrm{d}i_1}{\mathrm{d}t} + e_1 \\ u_2 &= Ri_2 + L\frac{\mathrm{d}i_2}{\mathrm{d}t} + e_2 \end{aligned} \tag{4}$$

Assume that machine speed is constant at steady state, solution will be simplified by replacing the time by the angle.

$$\begin{aligned} u_1 &= Ri_1 + L\omega\frac{\mathrm{d}i_1}{\mathrm{d}\theta} + e_1 \\ u_2 &= Ri_2 + L\omega\frac{\mathrm{d}i_2}{\mathrm{d}\theta} + e_2 \end{aligned} \tag{5}$$

Where: θ is angle of position of the rotor.

The phase-torque is proportional to a product of the phase current and phase emf, and disproportional to the rotor speed.

$$m_1 = \frac{i_1 e_1}{\omega} \qquad m_2 = \frac{i_2 e_2}{\omega} \tag{6}$$

Total motor torque is given by sum of the phase-torque.

$$m = m_1 + m_2 \tag{7}$$

The electromotive force induced in the stator coils can by expressed as a sinusoidal or co- sinusoidal function, which is lagged by the feeding voltage by angle γ. Angle γ depends from the torque of the machine.

$$e_1 = E.\sin(\theta - \gamma) \qquad e_2 = E.\cos(\theta - \gamma) \tag{8}$$

Where: E -is the maximal value of the induced electromotive force.

4 Converter Configuration

To form two-phase voltage supply system, was used the triac converter. The converter consists of the double secondary winding supply transformer set by two triacs on each of the phases. The triacs are controlled by a micro-controller like that way to constitute two phase voltage on the input motor terminal. The micro-controller must be on the mains voltage synchronised. The output waveform for each of the phases is given on the Figure 4.

The converter output voltage can be mathematically expressed by Fourier series as follows:

$$u_1 = \frac{8U_m}{\pi} \sum_{k=1}^{\infty} (-1)^{k+1} \frac{1}{4-(2k+1)^2} \sin\left[(2k+1)\omega t\right]$$
$$u_2 = \frac{8U_m}{\pi} \sum_{k=1}^{\infty} \frac{1}{4-(2k-1)^2} \cos\left[(2k+1)\omega t\right] \tag{9}$$

Where: U_m is a maximal value of the mains voltage; k is a positive integer with $k = 1, 2, 3\ldots$

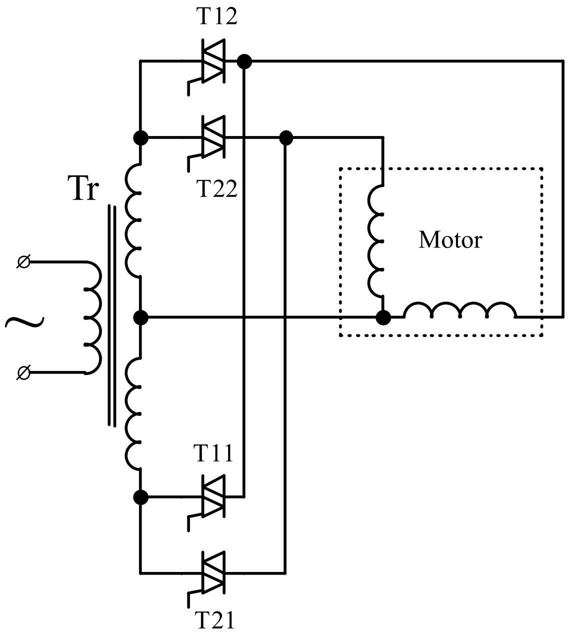


Figure 3
Triac converter circuit

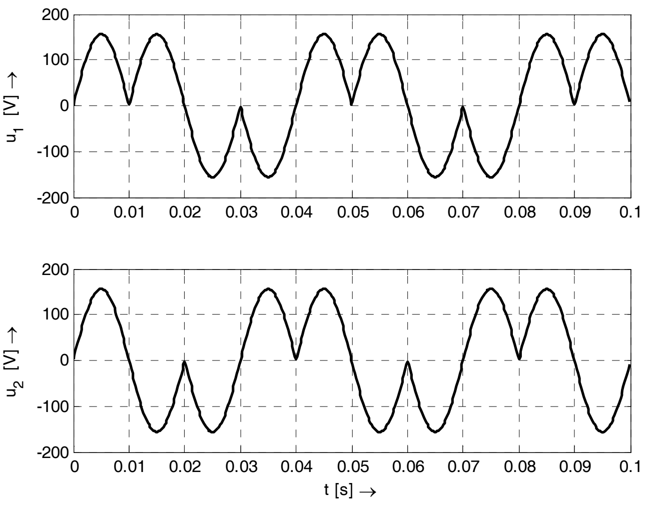


Figure 4
The converter output voltages waveform

The angle of rotor position θ is proportional to time t by expression:

$$\theta = \omega t \tag{10}$$

5 Analytical Solution of the Voltage Equation

To predict the phase curents waveform, it's necessary to solve the voltage differential equations (5). Substituting (8) and (9) into (5) we obtain:

$$\begin{aligned}
&\frac{8U_m}{\pi}\sum_{k=1}^{\infty}(-1)^{k+1}\frac{1}{4-(2k+1)^2}\sin\left[(2k+1)\theta\right] = Ri_1 + L\omega\frac{\mathrm{d}i_1}{\mathrm{d}\theta} + E\sin(\theta-\gamma)\\
&\frac{8U_m}{\pi}\sum_{k=1}^{\infty}\frac{1}{4-(2k-1)^2}\cos\left[(2k+1)\theta\right] = Ri_2 + L\omega\frac{\mathrm{d}i_2}{\mathrm{d}\theta} + E\sin(\theta-\gamma)
\end{aligned} \tag{11}$$

Presented linear differential equations have an analytical solution of the form:

$$\begin{aligned}
&i_1 = \frac{8U_m}{\pi\omega L}\sum_{k=1}^{\infty}\left\{\frac{(-1)^{k+1}}{4-(2k-1)^2}\cdot\frac{(\omega L)^2}{R^2+\left[\omega L(2k-1)\right]^2}\left[\frac{R}{\omega L}\sin\left[(2k-1)\theta\right]-(2k-1)\cos\left[(2k-1)\theta\right]\right]\right\}+\\
&\frac{ER}{R^2+(\omega L)^2}\cos(\theta-\gamma)+\frac{E\omega L}{R^2+(\omega L)^2}\sin(\theta-\gamma)+C.e^{-\frac{R}{\omega L}\theta}\\
&i_2 = \frac{8U_m}{\pi\omega L}\sum_{k=1}^{\infty}\left\{\frac{1}{4-(2k-1)^2}\cdot\frac{(\omega L)^2}{R^2+\left[\omega L(2k-1)\right]^2}\left[\frac{R}{\omega L}\cos\left[(2k-1)\theta\right]+(2k-1)\sin\left[(2k-1)\theta\right]\right]\right\}-\\
&\frac{ER}{R^2+(\omega L)^2}\sin(\theta-\gamma)+\frac{E\omega L}{R^2+(\omega L)^2}\cos(\theta-\gamma)+C.e^{-\frac{R}{\omega L}\theta}
\end{aligned} \tag{12}$$

For steady state: $\theta \to \infty$, $C.e^{-\frac{R}{\omega L}\theta} \to 0$;

6 Current and Torque Calculation

To calculate the currents and torque waveform, the following parameters were used:

- terminal supply voltage	$U = 110\,V$;
- resistance of the phase coil:	$R = 17.1\,\Omega$;
- inductance of the phase coil:	$L = 0,536\ H$

- stator induced voltage: $e = 45\,V$ for the speed of $\omega = 157\ rad/s$;

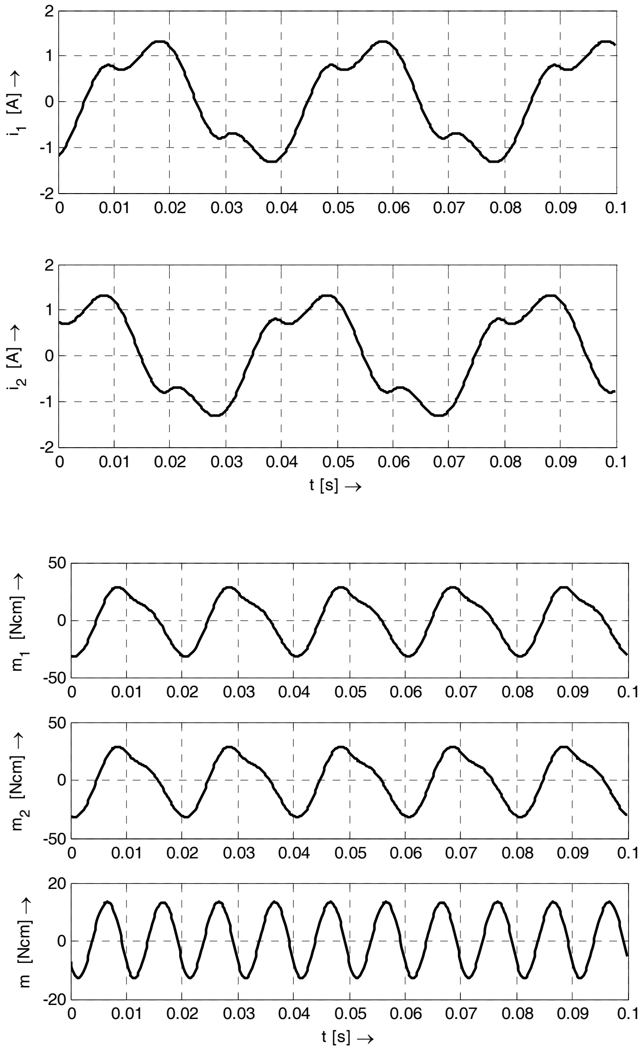


Figure 5
Plot of the currents and torque for no loaded machine

Figure 5 shows instantaneous values of the phase curents of the machine for steady state and no loaded machine. The speed of the machine is $157\ rad/s$.

There is shown of per-phase and total torque waveforms too. The torque ripple with amplitude about 15 *Ncm* is present in plot of total torque.

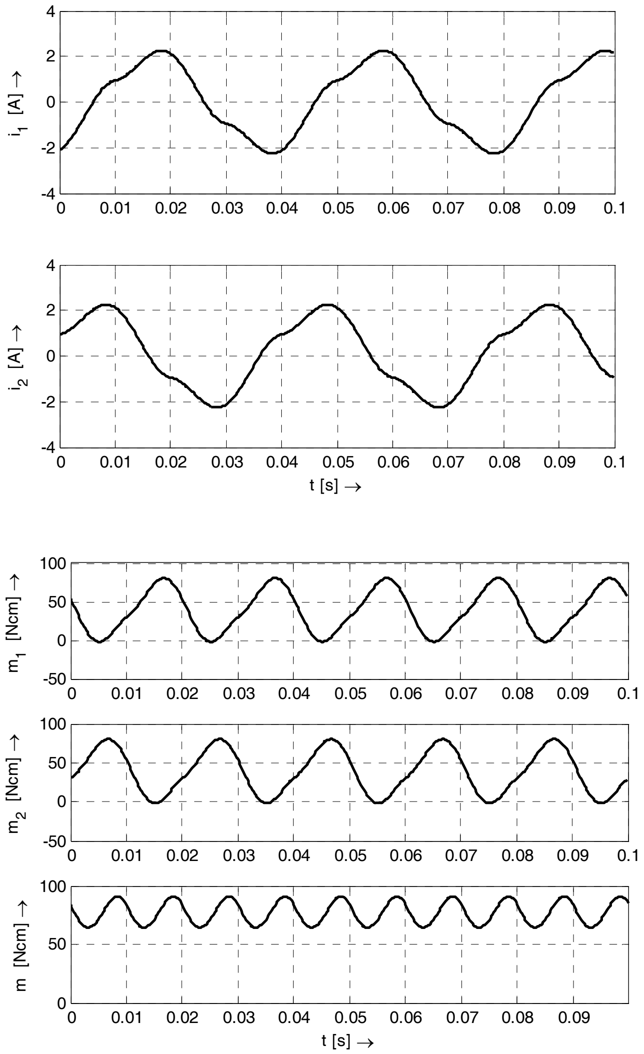


Figure 6
Plot of the currents and torque for loaded machine

Figure 6 shows instantaneous values of the phase curents of the machine for steady state and loaded machine. The machin is loaded by 80 *Ncm* torque. The second waveform of the Fig. 6 shows per-phase and total torque waveforms.

Conclusions

Steady-state performance of the permanent magnet synchronous motor, supplied by triac converter is shown. Equations, which enable to predict steady-state characteristics were developed. It was shown that the constant torque ripple in waveform of electromagnetic torque of the machine is present.

Acknowledgement

The financial support of the Science Grant Agency of the Slovakia, research grant No. 1/2178/05, is acknowledged.

References

[1] V. Hájek, H. Kuchyňková, “Losses Analysis and the Efficiency Optimization of the Automotive Electric Machines”; in: 14th International Conference on Electrical Drives and Power Electronics, pp. 133-135, 3-5 October 2001, High Tatras, Slovakia

[2] L. Klug, ”Brushless permanent magnet machine design and simulation”; in: 14th International Conference on Electrical Drives and Power Electronics, 3-5 October 2001, High Tatras, Slovakia

[3] L. Schreier, M. Chomát, I. Doležal, “Effect of machine geometry on higher harmonics contient in air-gap magnetic field of synchronous reluctance machine”; in: Scientific letters of Silesian University of Technology, z.176, pp. 259-266, Gliwice 2001, Poland

[4] P. Záskalický, M. Záskalická, “Behaviour of the Two-phase Permanent Magnet Synchronous Motor Supplied by Rectangular Voltage”; in: Acta Technica CSAV 50 (2005), 195-206, Prague, Czech Republic

Time and Memory Profile of a Process Functional Program

Ján Kollár, Jaroslav Porubän, Peter Václavík

Department of Computers and Informatics
Faculty of Electrical Engineering and Informatics
Technical University of Košice
Letná 9, 042 00 Košice, Slovakia
Jan.Kollar@tuke.sk

Abstract: An execution profiling attempts to provide feedback by reporting to the programmer information about inefficiencies within the program Instead of writing whole code highly optimized, the programmer can initially write simple maintainable code without much concern for efficiency. Profiling is an effective tool for finding hot spots in a program or sections of code that consumes most of the computing time and space. The paper presents already implemented execution profiler for process functional program. From the viewpoint of implementation, process functional language is between an impure eager functional language and a monadic lazy pure functional language. The key problem of execution profiling is to relate gathered information about an execution of the program back to the source code in well defined manner. The paper defines language constructs for monitoring resource utilization during the program execution. In our solution programmer can associate label with each expression in a program. All resources used during the evaluation of a labeled expression are mapped to the specified label. The paper is concerned with formal profiling model. Research results are presented on sample program illustrating different types of time and space profiles generated by already implemented profiler for process functional programs.

Keywords. Functional programming, program profiling, process functional language, formal profiling model

1 Introduction

A purely functional language is concise, composable and extensible. The reasoning about the pure functional programs defined in terms of expressions and evaluated without side effects is simpler than the reasoning about the imperative programs describing the tasteful systems. From the viewpoint of systems design, it seems more appropriate (at least to most of programmers) to describe the systems using an imperative language, expressing the state explicitly by variables as memory cells. Although the reliability of an imperative approach may be increased using object oriented paradigm, it solves neither the problem of reasoning about the functional correctness of fine grains of computation, since they are still affected by subsequent updating the cells in a sequence of assignments, nor the problem of profiling the program to obtain the execution satisfying the time requirements of a user.

Using the today compilers, code generators and tools, programmer can define functionality of a program on a higher abstract level than anytime before. Many programmers write their programs without knowledge of resource utilization during the program execution what leads to inefficiencies within the code. Barry Boehm reports that he has measured that 20 percent of the routines consume 80 percent of the execution time 0. Donald Knuth found that less than 4 percent of a program usually accounts for more than 50 percent of its run time 0. That is why the code optimization is so important. An execution profiling attempts to provide feedback by reporting to the programmer information about inefficiencies within the program 0. Informations about resource utilization are collected during the program execution. Instead of writing whole code highly optimized, the programmer can initially write simple, maintainable code without much concern for efficiency. Once completed the performance can be profiled, and effort spent improving the program where it is necessary 0. Profiling 00 is an effective tool for finding hot spots in a program, the functions or sections of code that consume most of the computing time. Profiles should be interpreted with care, however. Given the sophistication of compilers and the complexity of caching and memory effects, as well as the fact that profiling a program affects its performance, the statistics in a profile can be only approximate.

Many of ideas for process functional program profiling come out a pure functional program profiling because of the same functional basis 00,000,0,0. Our previous work proved that all process functional programs can be easily transformed into pure functional programs 0 using state transformers and monads. The paper presents our approach to profiling of process functional program and formal model of process functional program profiling. It is simple to extend the approach to both imperative and functional language.

2 Process Functional Language PFL

From the viewpoint of implementation, PFL is between an impure eager functional language and a monadic lazy pure functional language. The main difference between a process functional language and a pure functional language is variable environment which is designed to fulfill the needs of easier state representation in a functional program 0.

Variable environment in PFL is a mapping from variable to its value. The variable environment are updated and accessed during the runtime implicitly applying the process to values. The process in the process functional language differs from a function in a purely functional language only by its type definition. Let us define process *p* as an example.

$$p :: a\ Int \rightarrow b\ Int \rightarrow Int$$

$$p\ x\ y = x + y$$

Applying the process *p* to arguments, for example *p* 2 3, expression evaluates to 5, environment variable *a* is updated to value 2 and environment variable *b* is update to value 3. If the process is applied to a control value (), for example *p* () 4 than the process is evaluated using the current value of the environment variable *a* and variable *b* is updated to 4.

3 Execution Profiling

There are two main resources that are utilized in program and systems: computation time and memory space. Although it would be better to minimize both time and space, it is well understood that these two requirements are contradictory and it is impossible to fulfill both at the same time. Before being able to improve the efficiency of a program, a programmer must be able to 0:

- Identify execution bottlenecks of the program - parts of a program where much of time and space is used.
- Identify the cause of these bottlenecks

The potential benefits of execution profiling were first highlighted by Knuth 0. A profiler must conform two main criteria:

- must measure the distribution of the key program resources,
- measurement data must be related to the source code of a program in understandable manner.

Execution profile describes resource distribution during the program execution. Informations about resource distribution are gathered during the program execution. The profiling cycle describes the process of improving the program efficiency based on the program execution profile. The key problem of execution profiling is to relate gathered information about an execution of the program back to the source code in well defined manner. This is difficult when functional program is profiled since it provides higher level of abstractions than imperative one. Some features of a functional language, which makes program profiling more difficult than profiling an imperative program are: program transformation during compilation, polymorphism, higher-order function, lazy evaluation, a lot of simple functions within code.

4 Simple Program Profiling

Since our aim is "to compute" resource utilization at any point of computation, we define special constructs to monitor the resource utilization during the PFL program execution. In our solution programmer can associate *label* with each expression in a program as follows:

label *name e*

All resources used during the evaluation of an expression *e* are mapped to the center specified with label *name*. Using this construct programmer can concentrate on a specific part (or parts) of a program. Expression **label** *name e* is evaluated to value of *e*. Construct *label* is useful for the profiling purposes only. Of course, it is necessary to preserve the semantics of the expressions labeling during the transformation of the program when it is compiled. To be more precise, constructs for conditional profiling were incorporated into the process functional language. The first one is as follows.

label *name* e **when** e_c

If expression e_c is evaluated to value *True* of the *Bool* type, then all resources used during the evaluation of *e* are associated with label *name*. Otherwise, all used resources are attributed to the parent center. Of course, evaluation of e_c can not update the variable environment, because it is necessary to evaluate the program to the same value during the program profiling as during the program execution. On the other side, variable environment can be accessed during the evaluation of expression e_c. Fulfillment of this is rule checked during the static analysis in the compiler. Resources used during the evaluation of the conditional expression e_c are attributed to the special label *profiling* representing profiling overhead costs. All labeling inside the e_c are ignored.

It is clear, that conditional labeling is not the same as

if e_c **then label** *name e* **else** *e*

because of two main reasons:

- expression e_c is evaluated only during the profiling not the program execution,
- all resources used during the evaluation of e_c are attributed to the center with label *profiling* regardless of labeling in e_c.

Conditional profiling can enormously extend the time of profiling depending on the complexity of expressions e_c. Using conditional profiling labeled center can be dynamically activated based on decision during the execution of a program. Next example presents conditional labeling.

```
label "test" is_prime n when n > 100
```

5 Inheritance Profiles

Inheritance profiling can reduce the time spent with program profiling concentrating on smaller grains (pats of a program) than in simple profiling. Programmer can profile a part of program/function/expression depending on arguments and context. The usage of inheritance profile is explained on example. Usually the cost of function evaluation depends on arguments to which are function applied. Sometimes it is useful to consider the context of function in which it is called - parent. That is why the inheritance profiles are created.

On Figure 1 call graph of a simple program is depicted. Function *h* is called from function *f* 10 times with total cost 500 and from function *g* 20 times with total cost 100. Simple profile for the program is on Figure 2. Figure 3 presents inheritance profile for the presented call graph and function *h*. The first one is statistical profile which is generated from count and simple profile. The second one is measured accurate inheritance profile.

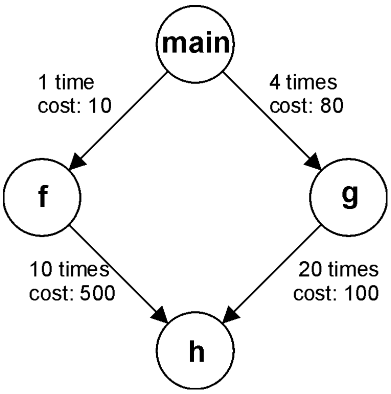


Figure 1
Call graph example

Function	Called	Cost
f	1	10
g	4	80
h	30	600

Figure 2
Simple profile

Parent→ Function	**Called Total**	**Cost Statistical Inheritance**	**Cost Accurate Inheritance**
f→h	10/30	200	500
g→h	20/30	400	100

Figure 3
Inheritance profile

In our profiler a few constructs for inheritance profile support have been implemented. The first one construct defined for conditional labeling with regard to parent context.

label *name e* **when enclosed** $name_c$

Using this construction, labeled center can be activated if parent center is same as specified. This construction can be used to create inheritance profiles. Resources used during the evaluation of expression *e* are attributed to the center with label *name* only if parent center is $name_c$. Otherwise, resources are attributed to the parent center. Next example presents usage of conditional enclosed labeling.

```
f = label "f" h 500

g = label "g" h 100

h n = label "f-h" (label "f-g" (p n)

    when enclosed "f") when enclosed "f"
```

For more flexible inheritance profiling two other constructions were defined.

label *name* **inherits** *e*

label *name* **inherits** *e* **when** e_c

Parent context are automatically added to the current labeled center. Inheritance profiling is not limited only to two levels parent/child.

Next example presents labeling for inheritance profiling of a simple process functional program.

```
f = label "f" h 500

g = label "g" h 100

h n= label "h" inherits (p n)
```

Function h can be evaluated from function f or h. Using the inheritance profiling label "h" is always connected with context of evaluation (function "f" or "g").

6 Formally Based Program Profiling

This section presents formal model of process functional program profiling. Our approach is presented on subset of PFL with constructs for profiling - simplified PFL. All PFL constructions are transformed to simplified PFL during the program compile time. This approach can be extended to all PFL language constructs. The meta-variables and categories for simplified PFL language are as follows:

$Prg \in$ Program $\quad Def \in$ Definition $\quad e \in$ Expression $\quad x \in$ Var

$f \in$ FncName $\quad \oplus \in$ Primitive $\quad y \in$ EnvVar $\quad C \in$ Constructor

$name, label \in$ Label

The meta-variables can be primed or subscripted. The syntactic category Primitive defines strict primitive operations like elementary arithmetic operations. The syntactic category Var represents variable identifiers and syntactic category EnvVar represents environment variable identifiers. The syntactic category Constructor comprises constructors of algebraic types. Primitive types, such as Int and Real, are included in the syntactic category Constructor as zero arity constructors. Syntactic category Label comprises label names. Program in simplified PFL consists of processes, functions and main expression which are evaluated during the program execution. Abstract syntax of simplified PFL is as follows.

$e ::= x$	Variable
$\mid\ f$	Function
$\mid\ e_1 \oplus e_2$	Primitive
$\mid\ y\,()$	Access
$\mid\ y\ e$	Update
$\mid\ C\ e_1 \ldots e_2$	Constructor
$\mid\ e_1\ e_2$	Aplication
$\mid\ \text{case } e \text{ of } \{C_i\ x_i \ldots x_{m_i} \to e_i\}_{i=1}^{n}$	Case
$\mid$ label *name e*	Label
$\mid$ label *name e* when e_c	CondLabel
$\mid$ label *name e* when enclosed $name_c$	EncLabel
$\mid$ label *name e* inherits *e*	InhLabel
$\mid$ label *name e* when *e* when e_c	InhCondLabel

The value v of an expression is either a lambda abstraction or a value of an algebraic type

$$v ::= C\ v_1 \ldots v_n$$
$$\mid \lambda x.e$$

where $n \geq 0$.

The runtime state is defined by environments env_v, env_e. Environment env_f is created during the compilation. Environment env_v represents the heap for the values of lambda variables and env_e is a set of memory cells for storing values of environment variables.

$$env_f \in \mathrm{Env_f} = \mathrm{FncName} \to \mathrm{Expr}$$
$$env_v \in \mathrm{Env_v} = \mathrm{Var} \to \mathrm{Value}$$
$$env_e \in \mathrm{Env_e} = \mathrm{EnvVar} \to \mathrm{Value}$$
$$(env_v, env_e) = s \in \mathrm{State} = \mathrm{Env_v} \times \mathrm{Env_e}$$

The semantic rules for simplified PFL expressions are defined on Figure 4. Figure Figure 5 presents semantic rules for label expressions. All rules are named corresponding to abstract syntax rule names. The predicate *matches* for pattern matching and operator *extract* for extracting *i*-th item value of the structure constructed by $C\ v_1\ ...\ v_i\ ...\ v_n$ are defined as follows.

$$matches\ v\ \ C\ x_1 \ldots x_i \ldots x_n \Leftrightarrow v = C\ v_1 \ldots v_i \ldots v_n$$
$$extract\ C\ v_1 \ldots v_i \ldots v_n\ \ i = v_i, \mathrm{where}\ 1 \leq i \leq n$$

The notation

$$env_f, label : \langle e, s \rangle \to_\mu (v, s')$$

defines that expression e is evaluated in environment env_f considering the state s and current label *label* and produces the value v, new state s' and resource environment μ. Resource environment maps label to resources used during the evaluation of labeled expression with specified label.

$$l_i \in \mathrm{Label}, \rho_\mathrm{i} \in \mathrm{Resources}, 1 \leq i \leq n$$
$$\mu \in \mathrm{ResourceMap} = \mathrm{Label} \to \mathrm{Resources}$$
$$\mu = [l_1 \to \rho_1] \ldots [l_n \to \rho_n]$$
$$(\mu_1 \cup \mu_2)\, l = \mu_1\, l + \mu_2\, l$$

The costs of elementary operation such as variable access or function application are defined as follows.

V	cost of variable access
F	cost of closure creation
P	cost of primitive operation evaluation
E_a	cost of environment variable access
E_u	cost of environment variable access
C	cost of constructor creation
A	cost of function application

D	cost of case evaluation
L	cost of lambda abstraction evaluation

7 Implementation

Implemented process functional program profiler nowadays supports five types of profiles:

1 frequency count profile

2 time profile

3 heap profile

4 maximum requirements heap profile

5 variable access/update profile

Program profile is created during the execution using the sampling method. Execution is interrupted in specified time intervals (predefined value is 10 milliseconds) and information about used resources are collected and attributed to the current labeled center. Program profiling increases execution time approximately from 5 to 10% depending on the concrete program and labeling. Formal semantics of the execution profiling is out of the scope of this paper and can be found in 0.

$$env_f, label : \langle x, (env_v, env_e) \rangle \rightarrow_{[label \mapsto \mathrm{V}]} (env_v\, x, (env_v, env_e)) \qquad \text{Variable}$$

$$\frac{env_f, label : \langle env_f\, f, s \rangle \rightarrow_{\mu} (v, s')}{env_f, label : \langle f, s \rangle \rightarrow_{\mu \cup [label \mapsto F]} (v, s')} \qquad \text{Function}$$

$$\frac{\begin{array}{l} env_f, label : \langle e_1, s \rangle \rightarrow_{\mu_1} (v_1, s_1) \\ env_f, label : \langle e_2, s_1 \rangle \rightarrow_{\mu_2} (v_2, s_2) \end{array}}{env_f, label : \langle e_1 \oplus e_2, s \rangle \rightarrow_{\mu_1 \cup \mu_2 \cup [label \mapsto P_{\oplus}]} (v_1 \oplus v_2, s_2)} \qquad \text{Primitive}$$

$$env_f, label : \langle y\,(), (env_v, env_e) \rangle \rightarrow_{[label \mapsto \mathrm{E}_a]} (env_e\, y, (env_v, env_e)) \qquad \text{Access}$$

$$\frac{env_f, label : \langle e, (env_v, env_e) \rangle \rightarrow_{\mu} (v, (env_v', env_e'))}{env_f, label : \langle y\ e, (env_v, env_e) \rangle \rightarrow_{\mu \cup [label \mapsto \mathrm{E}_u]} (v, (env_v', env_e'[y \mapsto v]))} \qquad \text{Update}$$

$$\frac{\begin{array}{l} env_f, label : \langle e_1, s \rangle \rightarrow_{\mu_1} (v_1, s_1) \\ \ldots \\ env_f, label : \langle e_i, s_{i-1} \rangle \rightarrow_{\mu_i} (v_i, s_i) \\ \ldots \\ env_f, label : \langle e_n, s_{n-1} \rangle \rightarrow_{\mu_n} (v_n, s_n) \end{array}}{env_f, label : \langle C\, e_1 \ldots e_i \ldots e_n, s \rangle \rightarrow_{\mu_1 \cup \ldots \cup \mu_i \cup \ldots \cup \mu_n \cup [label \mapsto \mathrm{C}]} (C\, v_1 \ldots v_i \ldots v_n, s_n)} \qquad \text{Constructor}$$

$$\frac{\begin{array}{l} env_f, label : \langle e_1, s \rangle \rightarrow_{\mu_1} (\lambda x.e, s') \\ env_f, label : \langle e_2, s' \rangle \rightarrow_{\mu_2} (v_2, (env_v, env_e)) \\ env_f, label : \langle e[\overline{x} / x], (env_v[\overline{x} \mapsto v_2], env_e) \rangle \rightarrow_{\mu_3} (v, s'') \end{array}}{env_f, label : \langle e_1\, e_2, s \rangle \rightarrow_{\mu_1 \cup \mu_2 \cup \mu_3 \cup [label \mapsto \mathrm{A}]} (v, s'')} \qquad \text{Application}$$

$$\frac{\begin{array}{l} env_f, label : \langle e, s \rangle \rightarrow_{\mu} (v', (env_v', env_e')) \\ matches\ v'\ (C_j\, x_1 \ldots x_{m_j}) \\ env_f, label : \langle e_j, (env_v' [\overline{x_1} \mapsto extract\ v'\, 1] \ldots \\ \qquad \ldots [\overline{x_{m_j}} \mapsto extract\ v'\, m_j], env_e' \rightarrow_{\mu_j} (v'', s'') \end{array}}{env_f, label : \langle \mathrm{case}\ e\ \mathrm{of}\ \{C_i\, x_1 \ldots x_{m_i} \rightarrow e_i\}_{i=1}^{n}, s \rangle \rightarrow_{\mu \cup \mu_j \cup [label \mapsto \mathrm{D}]} (v'', s'')} \qquad \text{Case}$$

$$env_f, label : \langle \lambda x.e, s \rangle \rightarrow_{[label \mapsto \mathrm{L}]} (\lambda x.e, s) \qquad \text{Lambda}$$

Figure 4
The semantics of expressions

$$\frac{env_f, name : \langle e, s\rangle \rightarrow_\mu (v, s')}{env_f, label : \langle \text{label } name\ e, s\rangle \rightarrow_\mu (v, s')} \qquad \text{Label}$$

$$\frac{\begin{array}{l} env_f, \text{profiling} : \langle e_c, s\rangle \rightarrow_{\mu_c} (v_c, s_c) \\ matches\ v_c\ True \\ env_f, name : \langle e, s\rangle \rightarrow_\mu (v, s') \end{array}}{env_f, label : \langle \text{label } name\ e \text{ when } e_c, s\rangle \rightarrow_\mu (v, s')} \qquad \text{CondLabel}^{\text{tt}}$$

$$\frac{\begin{array}{l} env_f, \text{profiling} : \langle e_c, s\rangle \rightarrow_{\mu_c} (v_c, s_c) \\ matches\ v_c\ False \\ env_f, label : \langle e, s\rangle \rightarrow_\mu (v, s') \end{array}}{env_f, label : \langle \text{label } name\ e \text{ when } e_c, s\rangle \rightarrow_\mu (v, s')} \qquad \text{CondLabel}^{\text{ff}}$$

$$\frac{\begin{array}{l} name_c = label \\ env_f, name : \langle e, s\rangle \rightarrow_\mu (v, s') \end{array}}{env_f, label : \langle \text{label } name\ e \text{ when enclosed } name_c, s\rangle \rightarrow_\mu (v, s')} \qquad \text{EncLabel}^{\text{tt}}$$

$$\frac{\begin{array}{l} name_c \neq label \\ env_f, label : \langle e, s\rangle \rightarrow_\mu (v, s') \end{array}}{env_f, label : \langle \text{label } name\ e \text{ when enclosed } name_c, s\rangle \rightarrow_\mu (v, s')} \qquad \text{EncLabel}^{\text{ff}}$$

$$\frac{env_f, context(label, name) : \langle e, s\rangle \rightarrow_\mu (v, s')}{env_f, label : \langle \text{label } name \text{ inherits } e, s\rangle \rightarrow_\mu (v, s')} \qquad \text{InhLabel}$$

$$\frac{\begin{array}{l} env_f, \text{profiling} : \langle e_c, s\rangle \rightarrow_{\mu_c} (v_c, s_c) \\ matches\ v_c\ True \\ env_f, context(label, name) : \langle e, s\rangle \rightarrow_\mu (v, s') \end{array}}{env_f, label : \langle \text{label } name \text{ inherits } e \text{ when } e_c, s\rangle \rightarrow_\mu (v, s')} \qquad \text{InhCondLabel}^{\text{tt}}$$

$$\frac{\begin{array}{l} env_f, \text{profiling} : \langle e_c, s\rangle \rightarrow_{\mu_c} (v_c, s_c) \\ matches\ v_c\ False \\ env_f, label : \langle e, s\rangle \rightarrow_\mu (v, s') \end{array}}{env_f, label : \langle \text{label } name \text{ inherits } e \text{ when } e_c, s\rangle \rightarrow_\mu (v, s')} \qquad \text{InhCondLabel}^{\text{ff}}$$

Figure 5
The semantics of label expressions

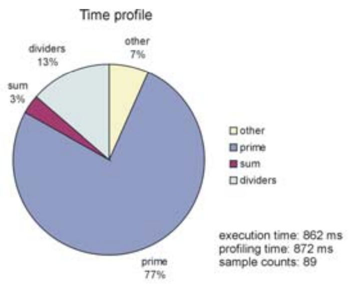
The next section presents simple example with profiling outputs from the implemented profiler for process functional language. The problem solved by the program is to

1 generate prime numbers from 1 to 100 (label *prime*),
2 sum prime number from 1 to 100 (label *sum*),
3 calculate dividers of the sum (label *dividers*),
4 generate list of values from 1 to 1000 (without any label).

PFL program source code for the problem with profile labeling is as follows.

```
gen_list m n = if m > n then [] else m : gen_list (m + 1) n

can_be_divided n m = (n % m) == 0

is_prime_number n = not (foldl (or) False (map (can_be_divided n)

                         (gen_list 2 (n - 1))))

prime_numbers from to = filter (is_prime_number) (gen_list from to)

dividers n = filter (can_be_divided n) (gen_list 1 n)

main = (toUnit (label "dividers" dividers (

         label "sum" sum

           (label "prime" primeNumbers 1 100))))

  `bl` (toUnit (generateIntegerList 1 1000))
```

Time and memory profile for example program produced by the profiler is on Figure 6.



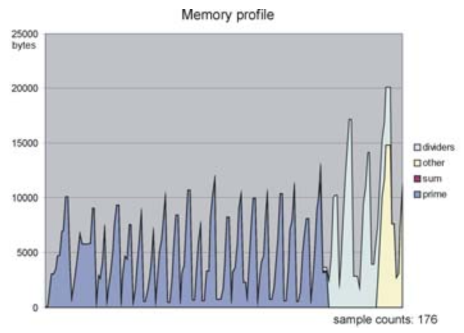


Figure 6
Time and memory profile

Conclusions

Using the execution profile of a program a programmer had to answer next two questions:

- How are resources distributed during the program execution?
- What is the effect of a particular modification of a program?

Our solution to process functional program execution profiling was presented in this paper. Using our method every expression in the PFL program can be separately profiled. The definition of profiling grains is up to the programmer. Suggested formal model can be used for reasoning about program profiling.

This work is based on our previous research of profiling and static evaluation of process functional programs 0. As a result, the static evaluation method is strongly associated with the source specification. This may help to a programmer while program development considering not just the function but also the behavior, represented by resources used. Combining execution profiling with static analysis look very promising in gathering information about resource utilization during program execution.

In the past, we have PFL-to-Java and PFL-to-Haskell generators developed. The subject of our current research is integrating aspect and process functional paradigm of programming. Our future plan is to extend profiling tools to object oriented PFL and to formal specification of a program profiling for parallel environment.

References

[1] B. W. Boehm: Improving Software Productivity. IEEE Computer 20, Vol. 9, 1987, pp. 43-57

[2] Ch. D. Clack, S. Clayman, D. Parrott: Lexical Profiling: Theory and Practice. Journal of Functional Programming Vol. 5, No. 2, 1993, pp. 225-277

[3] D. Hamlet: On subdomains: Testing, profiles, and components, Proceedings of the International Symposium on Software Testing and Analysis, Portland, Oregon, United States, August 21-24, 2000, pp.71-76

[4] J. Kollár: Partial Monadic Approach in Process Functional Language. Acta Electrotechnica et Informatica No. 1, Vol. 3, Košice, Slovak Republic, 2003, pp. 36-42

[5] J. Kollár, J. Porubän, P. Václavík, M. Vidiščák: Lazy State Evaluation of Process Functional Program. Proceding of 5th International Conference ISM 2002, Rožnov pod Radhoštem, Czech Republic, April 22-24, 2002

[6] D. E. Knuth: An Empirical Study of FORTRAN Programs. Software - Practice and Experience 1, 1971, pp. 105-133

[7] J. Porubän: Time and space profiling for process functional language. Proceeding of the 7th Scientific Conference with International Participation EMES '03, Felix Spa-Oradea, May 29-31, 2003, pp. 167-172

[8] N. Rojemo: nhc - Nearly a Haskell compiler, in Proceedings of La Wintermote, Dept of Computer Science, Chlamers University, Sweden, January 1994

[9] C. Runciman, D. Wakeling: Heap Profiling of Lazy Functional Programs. Journal of Functional Programming, Vol. 3, No. 2, pp. 217-245, 1993

[10] P. M. Sansom: Execution profiling for non-strict functional languages. Research Report FP-1994-09, Dept. of Computing Science, University of Glasgow, September 1994

[11] P. M. Sansom, S. L. Peyton Jones: Profiling lazy functional programs. Functional Programming, Glasgow 1992, Springer Verlag, Workshops in Computing, 1992

[12] P. M. Sansom, S. L. Peyton Jones: Time and space profiling for non-strict, higher-order functional languages, Proceedings of the 22nd ACM SIGPLAN-SIGACT symposium on Principles of programming languages, San Francisco, California, United States, January 23-25, 1995, pp. 355-366

[13] P. M. Sansom , S. L. Peyton Jones: Formally based profiling for higher-order functional languages, ACM Transactions on Programming Languages and Systems (TOPLAS), Vol. 19, No. 2, March 1997, pp. 334-385

[14] S. Rubin, R. Bodík , T. Chilimbi: An efficient profile-analysis framework for data-layout optimizations, Proceedings of the 29th ACM SIGPLAN-SIGACT symposium on Principles of programming languages, Portland, Oregon, January 16-18, 2002, pp. 140-153

[15] P. Wadler, P. Thiemann: The marriage of effects and monads. ACM Transactions on Computational Logic, Volume 4, Issue 1, 2003, pp. 1-32

Embedded Fuzzy Controller for Industrial Applications

Ferenc Farkas, Sándor Halász

Department of Electric Power Engineering, Budapest University of Technology and Economics, ferenc.f.farkas@ericsson.com

Abstract: The concept of the fuzzy logic makes feasible the creation of fuzzy controllers with low cost 16 bit microcontroller having the same performance as of controllers realized with more expensive Digital Signal Processor (DSP). In this article the implementation of such a fuzzy controller is proposed for 16 bit microcontroller with fast fuzzyfication-inference-defuzzyfication algorithm. Because the microcontroller receives information from the process via Analog-Digital Converter(s) and controls the process with the help of Digital-Analog Converter(s) the implemented algorithm does not use floating-point operations, only integer ones. However, for some type of fuzzy controllers, the error made by this algorithm is not greater than the error of a DSP based floating point algorithm.

Keywords: fuzzy logic, microcontroller, embedded systems

1 Introduction

1.1 Fuzzy Controller and Embedded Systems

The world of embedded control is experiencing a push into the realm of fuzzy logic. Even household machines are advertised as being intelligent with the help of the built-in fuzzy logic. The popularity of the fuzzy logic is due to its simplicity and effectiveness in solving control problems. Conference proceedings and related periodicals contain myriads of articles presenting the advantages of control systems using fuzzy logic. Although fuzzy controllers are not able to solve every control problems, and have some disadvantages as well [7], one of the main disadvantage of using the fuzzy controller in embedded systems is the great number of floating point calculations made in real-time. This huge calculation capacity requires the use of Digital Signal Processor (DSP), which is more expensive compared to a 16 bit microcontroller (μC).

In the past, manufacturers have contended with the performance versus cost tradeoffs with no apparent fulfillment of both. Although, in the last decade the complexity of DSPs has evolved while their price decreased, manufacturers are always interested in cost reduction due to permanent competition. In this article a short comparison is presented between a DSP and a microcontroller based fuzzy controller, pointing out that – depending on the type of the fuzzy controller used and the precision of the Analog-Digital Converters (ADC) and Digital-Analog Converters (DAC) built in the system, – in most of the cases the DSP based fuzzy controller is not able to outperform the microcontroller based counterpart.

1.2 Microcontroller versus DSP

The main features of embedded systems are the compact realization, robustness, and cheapness. Cheapness can be achieved by using low performance microcontroller connected to low capacity memory. Naturally, such a system cannot be compared to a more complex DSP from the calculation capacity point of view. Thus, microcontrollers can be used in limited applications, where floating-point calculations are not required, or their use is limited. At first sight, microcontrollers are not suitable for realizing fuzzy controllers, due to hundreds of floating-point arithmetic done in real time. This short come might be overcome with a look-up table, storing the response of the fuzzy controller for different input/output combinations. However, the memory capacity is a strong limitation, so the table dimension is. Thus, only a limited number of input(s)/output(s) pairs are stored in the table, and interpolation is used in between. This solution has two major drawbacks: 1) the interpolation is not a good approximation for nonlinear functions, and the table dimension limits the number of useful pairs stored; 2) the fuzzy controller is rigid, cannot adjust its parameter to the changing environment as it is proposed in [1]-[2].

The concept of fuzzy logic makes feasible the use of a fuzzy controller built on low cost 8 or 16 bit microcontroller for some applications. Manufacturers, like MOTOROLA, have recognized the power of fuzzy logic and have created fuzzy kernels and support tools for a number of their 8 bit and 16 bit microcontrollers. However, these support tools lack the generalization and mathematical reasoning.

Although DSP is mostly applied in those applications where huge floating-point operations are performed in real time, its high price does not help the spreading of fuzzy controller in mass production. On the other hand a conventional microcontroller (like the INTEL's 80186 microcontroller) can be bought for a very low price. The latter does not support directly the floating-point operations, but does support the operations of integer type, like addition, subtraction, division and multiplication. For this reason, one must think about an algorithm, which incorporates only integer operands.

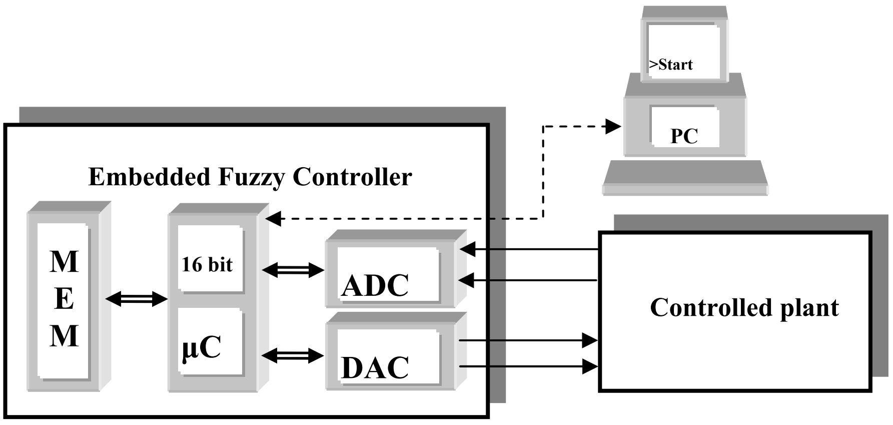


Figure 1
The concept of embedded fuzzy controller with ADC and DAC

In industrial applications, such as motion control, electrical drives, temperature and humidity stabilization, the fuzzy controller receives information from the controlled process via ADC(s) and controls it through DAC(s) as in Figure 1. For this reason, using a DSP in such cases is not far from the idea of making a fuzzy controller seeming better, faster and more accurate than it really is. Because the precision of ADC and DAC is always less than or equal to 16 bits, the use of 16 bit operands seems to be a reasonable compromise between accuracy and speed. Thus, the value stored in such an operand will be in the range of [0.65535]. Moreover, using appropriate fuzzy operations, the error made by the proposed algorithm is comparable to the DSP based fuzzy controller.

2 Theoretical Considerations

2.1 Starting from the Basic Idea

Let's consider two continuous and closed intervals (a,b), (E,F) on which the Euclidean distances are defined. For arbitrary $x \in (a,b)$, and $X \in (E,F)$ the following relation is held:

$$\frac{x-a}{b-a} = \frac{X-E}{F-E}\,. \tag{1}$$

Let's define the two intervals as being:

$$\begin{cases}(a,b) \equiv (0.0,1.0) \\ (E,F) \equiv (0,MAXINT), MAXINT = 65535\end{cases}, \tag{2}$$

that is, the (a,b) interval represents the real numbers between 0.0 and 1.0, while the $(0,MAXINT)$ interval represents the integer numbers from 0 to $MAXINT \equiv 2^{16}$. Because (E,F) interval defined in this way is not continuous, there is no one-to-one mapping anymore. That is, $x' \in (a,b)$ being defined closed enough to $x \in (a,b)$ in terms of Euclidean distance, x' will be mapped on the same $X \in (E,F)$ as x. The following notation will be used in the rest of this article: with lower case real numbers, while with capital letters integer numbers are denoted.

The following relations are obtained by rearranging relation (1), taking into account the definitions of the intervals, and replacing x with x', or x with y', Y with X:

$$x' = \frac{X}{MAXINT}, \tag{3}$$

$$Y = [y' \cdot MAXINT], \tag{4}$$

where $[x]$ represents the integer part of x. These relations show how a real value can be mapped on an integer one, and vice versa.

It is by no surprise, that the defined $(a,b) \equiv (0.0,1.0)$ interval is the input/output of the fuzzy controller, while the $(E,F) \equiv (0,MAXINT)$ interval is the input/output of the 16 bit AD/DA converters. Mapping of real value to integer and vice versa is performed by the ADC and DAC, respectively. However, the fuzzy controller implemented on a DSP requires real number(s) for its input(s), and outputs real

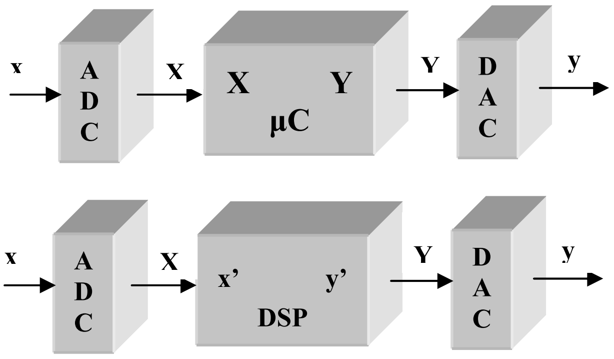


Figure 2
Microcontroller versus DSP. Which one is more suitable for an application?

number(s), as well. Thus, an algorithm for the DSP must convert the integer obtained from the ADC before applying to the fuzzy controller. Another algorithm should also convert the real number obtained from the output of the fuzzy controller to an integer one before feeding the DAC with the proper value. If it is so, there are two important questions: 1) what is the gain from using floating-point operands and operators? 2) can be implemented a fuzzy controller just using integer operands and operators?

In Figure 2 it is shown the two alternative fuzzy controllers: the upper one implemented on a microcontroller (μC), while the lower one on a DSP. The x value obtained from the controlled plant is converted by the ADC to X, which is directly used by the fuzzy controller implemented on a microcontroller. However, the DSP needs to convert this X value to another real value x' using relation (3). It is obvious, that generally $x \neq x'$, that is, the input value is not equal to the input of the fuzzy controller. The fuzzy controller implemented on the DSP creates the output value y', which is converted by another algorithm using relation (4) to Y value. This Y value (obtained directly from the microcontroller based fuzzy controller) is further converted to a real value y by DAC, and this y serves as a control signal. Again, it can be stated that generally $y' \neq y$, that is, the control signal is not equal to the output of the fuzzy controller. This gives a hint that floating-point calculation might be useless, because of the presence of AD and DA converters.

2.2 Error of the Integer Operators

Before diving into the deep water, it should be analyzed the behavior of the integer multiplication and division. Let's consider two arbitrary input values X_1, X_2 converted by the ADCs, and an output Y supplied by the controller which is fed to the DAC. The output of the multiplication operation is

$$Y = [y' \cdot MAXINT] = [(x_1' \cdot x_2') \cdot MAXINT], \tag{5}$$

and using relation (3) the output is obtained in function of the inputs:

$$Y = \left[\left(\frac{X_1}{MAXINT} \cdot \frac{X_2}{MAXINT} \right) \cdot MAXINT \right] = \left[\frac{X_1 \cdot X_2}{MAXINT} \right]. \tag{6}$$

Similar result is obtained for the integer division, when the output is

$$Y = [y' \cdot MAXINT] = [(x_1' / x_2') \cdot MAXINT], \tag{7}$$

and using relation (4) the output in function of the inputs is obtained:

$$Y = \left[\left(\frac{Y_1}{MAXINT} / \frac{Y_2}{MAXINT} \right) \cdot MAXINT \right] = \left[\frac{Y_1 \cdot MAXINT}{Y_2} \right]. \tag{8}$$

It is important to note, that always the multiplication in the numerator must be performed firstly, and the obtained 32 operand should be divided by the 16 bit denominator (these operations are directly supported by the INTEL 80186 microcontroller using the MUL and DIV opcodes). The obtained result consists of the integer part of the division and the 16 bit remainder. Because only the integer part of the division must be fed to the DAC, there is no difference between using a DSP with floating-point operators, or a microcontroller with integer operators. Similar result is obtained when the weighted average is calculated (combination of multiplication and division):

$$Y = [y' \cdot MAXINT] = \left[\frac{a \cdot x_1' + b \cdot x_2'}{a+b} \cdot MAXINT \right], \tag{9}$$

$$Y = \left[\frac{a \cdot MAXINT \cdot x_1' + b \cdot MAXINT \cdot x_2'}{a \cdot MAXINT + b \cdot MAXINT} \cdot MAXINT \right], \tag{10}$$

$$Y = \left[\frac{A \cdot x_1' \cdot MAXINT + B \cdot x_2' \cdot MAXINT}{A+B} \right] = \left[\frac{A \cdot X_1 + B \cdot X_2}{A+B} \right]. \tag{11}$$

It can be seen, that only for one multiplication, division or a combination of them the error of the integer operations is not propagated through the DAC converter. In the next subchapter, an investigation for the whole fuzzy controller is performed.

2.3 Error Propagation through the Fuzzy Controller

2.3.1 Fuzzyfication

One of the most common used fuzzy controller is the Mamdani type controller, with MIN-MAX operators, and triangle or trapezoidal membership functions. Let's denote $\{a_x^i, b_x^i, c_x^i, d_x^i\}$ the parameters of the i th input membership function, while $\{a_y^j, b_y^j, c_y^j, d_y^j\}$ the parameters of the j th output membership function. In case of triangle membership function it can be simply considered $b_x^i = c_x^i$ or $b_y^j = c_y^j$. The simplest fuzzyfication is the singleton one, when the fuzzyfication function is the identity function, e.g. *f(y)=y*. That means that input variables are the singleton fuzzy inputs. The membership value $\mu^i(x)$ of the input x corresponding to a membership function defined by the $\{a_x^i, b_x^i, c_x^i, d_x^i\}$ parameters is obtained by relation (12):

$$\mu^i(x) = \begin{cases} 0, x \le a_x^i \vee x \ge d_x^i \\ \dfrac{x - a_x^i}{b_x^i - a_x^i}, x < b_x^i \\ 1, b_x^i \le x \le d_x^i \\ \dfrac{d_x^i - x}{d_x^i - c_x^i}, c_x^i < x < d_x^i \end{cases} . \tag{12}$$

Let's consider an input x, where $a_x^i < x < b_x^i$, and calculate the $\mu^i(x)$ membership value for both floating-point and integer operators:

$$Y = \mu(X) = [\mu(x) \cdot MAXINT] = \left[\frac{x - a_x^i}{b_x^i - a_x^i} \cdot MAXINT \right], \tag{13}$$

$$Y = \mu(X) = \left[\frac{x \cdot MAXINT - a_x^i \cdot MAXINT}{b_x^i \cdot MAXINT - a_x^i \cdot MAXINT} \cdot MAXINT \right] = \left[\frac{(X - A_x^i) \cdot MAXINT}{B_x^i - A_x^i} \right] \tag{14}$$

where $a_x^i, b_x^i \in [0.0, 1.0]$ and $A_x^i, B_x^i \in [0, MAXINT]$ represent the real, respective integer parameters of the i th input membership function. In similar way should be calculated the $\mu^i(x)$ membership value for inputs with $c_x^i < x < d_x^i$. It was assumed in relation (14) that $a_x^i \cdot MAXINT = A_x^i$ and $b_x^i \cdot MAXINT = B_x^i$, which is not always true. In equation (3) the remainder of the division is not zero, that is, $a_x^i = \dfrac{A_x^i}{MAXINT} + a_{x \cdot RES}^i$, where $a_{x \cdot RES}^i$ is the remainder. In order to avoid any error, $a_x^i \cdot MAXINT = A_x^i$ must be held. This might look a restriction at first sight, but even in case of floating-point operands, the parameters of the membership functions are chosen to 2-3 places of decimals. Even with this restriction the parameters of the membership functions can be set for $MAXINT + 1 = 65536$ different value, which is enough for most of the applications. In conclusion, the fuzzyfication and the calculation of the membership value do not cause additional rounding error.

2.3.2 Inference

When the fuzzy controller has more then one input, generally the rule base is constructed in such way that AND relation exist between the rules. This AND rule is performed by the MIN operator in case of Mamdani type controller. That means, the "firing" degree of the k th rule is given by the following relation:

$$Y = \lambda(k) = [MIN\{\mu(x_1), \mu(x_2)\} \cdot MAXINT], \tag{15}$$

from where it can be concluded that there is no additional rounding error.

However, there are types of fuzzy controller where the AND relation is performed by multiplication. In those cases it is not so simple to decide what rounding errors one might have. Let's take two inputs, one belonging to the first, the other one belonging to the second membership function. Then the "firing" degree of the k th rule is

$$Y = \lambda(k) = [\mu(x_1) \cdot \mu(x_2) \cdot MAXINT] = \left[\frac{X_1 - A_x^1}{B_x^1 - A_x^1} \cdot \frac{X_2 - A_x^2}{B_x^2 - A_x^2} \cdot MAXINT \right], \qquad (16)$$

where both the numerator and the denominator is multiplied by $MAXINT$. There are two alternative solutions. An algorithm should be implemented in the first case which is able to divide a 48 bit number by a 32 bit number:

$$Y = \lambda(k) = \left[\frac{(X_1 - A_x^1) \cdot (X_2 - A_x^2) \cdot MAXINT}{(B_x^1 - A_x^1) \cdot (B_x^2 - A_x^2)} \right]. \qquad (17)$$

Although, this kind of algorithm has no additional rounding error – the 48 bit is divided by the 32 bit number, but only the integer part must be considered –, the running time of the algorithm might be significant for some applications. Another solution is also presented, which runs faster, but has rounding error. Let's multiply in equation (16) both the numerator and denominator with $MAXINT$:

$$Y = \lambda(k) = \left[\frac{\left(\frac{(X_1 - A_x^1) \cdot MAXINT}{B_x^1 - A_x^1} \right) \cdot \left(\frac{(X_2 - A_x^2) \cdot MAXINT}{B_x^2 - A_x^2} \right)}{MAXINT} \right] = \left[\frac{\mu(X_1) \cdot \mu(X_2)}{MAXINT} \right] \qquad (18)$$

That means, membership values are calculated individually using relation (12), then the obtained membership values are multiplied together and the resulted product is divided by $MAXINT$. Although, this simplified algorithm is supported by the MUL and DIV opcodes of the microcontroller, rounding error is introduced. This is due to the fact, that the two members in the numerator are not calculated precisely, only the integer part is taken into account (this is equivalent with one replacing the round brackets with brackets in the numerator). The rounding error can be estimated if one considers P_1, P_2, Q_1, Q_2 arbitrary integer numbers, and calculates the product of their quotient:

$$\begin{cases} \frac{Q_1}{P_1} \cdot \frac{Q_2}{P_2} = \frac{N_1 P_1 + R_1}{P_1} \cdot \frac{N_2 P_2 + R_2}{P_2} = \left(N_1 + \frac{R_1}{P_1} \right) \left(N_2 + \frac{R_2}{P_2} \right) \\ \frac{Q_1}{P_1} \cdot \frac{Q_2}{P_2} = N_1 N_2 + N_1 \frac{R_2}{P_2} + N_2 \frac{R_1}{P_1} + \frac{R_1}{P_1} \cdot \frac{R_2}{P_2} \end{cases}, \qquad (19)$$

where N_1, N_2 are the integer parts of the quotients and R_1, R_2 are the remainders.

It can be concluded from relation (19) that only the $N_1 \cdot N_2$ product is considered in equation (18), the rest three terms are omitted. The following inequality is held for the last three terms of relation (19):

$$N_1 \cdot \frac{R_2}{P_2} + N_2 \cdot \frac{R_1}{P_1} + \frac{R_1}{P_1} \cdot \frac{R_2}{P_2} < (MAXINT - 1) + (MAXINT - 1) + 1 = 2 \cdot MAXINT - 1 \text{,} \quad (20)$$

because both R_1, R_2 are less than $MAXINT$. From relation (18) it follows that additional rounding error occurs only if the following condition is held:

$$R_\mu + N_1 \cdot \frac{R_2}{P_2} + N_2 \cdot \frac{R_1}{P_1} + \frac{R_1}{P_1} \cdot \frac{R_2}{P_2} \geq MAXINT \text{,} \quad (21)$$

where $R_\mu = \mu(X_1) \cdot \mu(X_2) - N_\mu \cdot MAXINT$ is the remainder of division of the equation (18). From equation (18) and inequality (20) it can be concluded that the rounding error is between 0 and 2 bits. 0 bit error occurs when inequality (21) is not held, that is, when R_1, R_2 remainders are far less than P_1, P_2 and R_μ is also minor. 2 bit error occurs only and only if equality $R_\mu + N_1 \cdot \frac{R_2}{P_2} + N_2 \cdot \frac{R_1}{P_1} + \frac{R_1}{P_1} \cdot \frac{R_2}{P_2} = 2 \cdot MAXINT$ holds. However, this is a rear situation. Thus, it can be concluded that the average additional rounding error is 1 bit. Moreover, this 1 bit rounding error can even be absorbed by the imprecision of the DAC, taking into account that common DACs have only ±½ bit precision at conversion. Important to note, that since the 2 least significant bits are not passed to the 14 bit DAC, even the 2 bit error is not present at the output of the DAC.

MIN operator is used for the inference operator, as well, although there are fuzzy controllers where the multiplication is used instead. In case of MIN operator the height of the j th output membership function is defined by the minimum of the “firing” degree of the rules containing the same antecedents:

$$Y = h_y^j = [MIN\{\lambda(i), \lambda(k)\} \cdot MAXINT] \text{,} \quad (22)$$

which does not introduce additional rounding error. However, this is not true when multiplication is used instead of MIN operator. In case of multiplication the height of the j th output membership function is defined by the product of the “firing” degree of the rules containing the same antecedents:

$$Y = h_y^j = [\lambda(i) \cdot \lambda(k) \cdot MAXINT] \text{.} \quad (23)$$

There two possibilities: the $\lambda(i), \lambda(k)$ ”firing” degrees either have been calculated with MIN operator using equation (15) or with multiplication operator as in case of relation (16). In the former case

$$Y = h_y^j = [MIN\{\mu_i(x_1), \mu_i(x_2)\} \cdot MIN\{\mu_k(x_1), \mu_k(x_2)\} \cdot MAXINT] \text{,} \quad (24)$$

from where, without loosing the generality, it is considered that $\mu_i(x_1) < \mu_i(x_2)$ and $\mu_k(x_1) > \mu_k(x_2)$, from where it is obtained:

$$Y = h_y^j = [\mu_i(x_1) \cdot \mu_k(x_2) \cdot MAXINT]. \tag{25}$$

It is obvious, that equation (25) looks like equation (16), and thus, the same conclusion can be drawn: the additional rounding error is between 0 and 2 bits, and the average rounding error is 1 bit.

The situation is much complicated when the overall “firing” degree is calculated by multiplication. In that case the height of the j th output membership function is

$$Y = h_y^j = [(\mu_i(x_1) \cdot \mu_i(x_2)) \cdot (\mu_k(x_1) \cdot \mu_k(x_2)) \cdot MAXINT], \tag{26}$$

from where it is obtained by substituting the membership values

$$Y = h_y^j = \left[\frac{X_1 - A_x^i}{B_x^i - A_x^i} \cdot \frac{X_2 - A_x^i}{B_x^i - A_x^i} \cdot \frac{X_1 - A_x^k}{B_x^k - A_x^k} \cdot \frac{X_2 - A_x^k}{B_x^k - A_x^k} \cdot MAXINT \right]. \tag{27}$$

One might construct an algorithm which is able to calculate the 80 bit product of the numerator, which is divided by the 64 bit product of the denominator, in which case there is no rounding error. However, such algorithms will slower the inference of the fuzzy controller, which might not be suitable for some applications. For this reason, the following solution is proposed:

$$Y = h_y^j = \left[\frac{\lambda(i) \cdot \lambda(k)}{MAXINT} \right], \tag{28}$$

where $\lambda(i), \lambda(k)$ “firing” degrees are calculated using the equation (18). In this case additional rounding error exists. In order to find out the magnitude of the rounding error, let’s consider the worst case when both “firing” degrees have been calculated with 2 bit error.

That means, the $\lambda(i) \cdot \lambda(k)$ product should be replaced by $(\lambda(i)+2) \cdot (\lambda(k)+2) = \lambda(i) \cdot \lambda(k) + 2\lambda(i) + 2\lambda(k) + 4$ in order to calculate the precise value. This gives an additional rounding error if $R_\lambda + 2\lambda(i) + 2\lambda(k) + 4 \geq MAXINT$ inequality holds, where $R_\lambda = \lambda(i) \cdot \lambda(k) - N_\lambda \cdot MAXINT$ represents the remainder of the division from relation (28). The largest rounding error occurs when both “firing” degree is equal to $MAXINT$, and $R_\lambda \geq MAXINT - 4$ inequality holds. Although this rounding error of 5 bits seems very large and might not be acceptable, in common applications never occur. However, 4 bit errors might still persist, and the average rounding error is around 2 bits, taking into account that the $\lambda(i), \lambda(k)$ “firing” degrees are calculated with 1 bit rounding error in average. This 2 bit error might still bother the designer of a fuzzy controller. However, 14 bit DACs are used in a large number of applications, in which case even 4 bit rounding error does not appear at the output of the 14 bit DAC, since the 2 least significant bits are not passed to the 14 bit DAC.

The aggregation of the output membership functions can be done in several ways [4]-[6]. The two most popular aggregations are the MAX operator and the bounded sum. When MAX operator is used for aggregating the output membership function

$$Y_{fuzzy} = \left[\underset{j}{MAX} \{h_y^j\} \right] \tag{29}$$

does not contain additional rounding error. So it is, when aggregation is calculated with the bounded sum operator:

$$Y_{fuzzy} = \left[\left\langle \sum_j h_y^j \right\rangle_{\leq MAXINT} \right]. \tag{30}$$

Thus, it can be concluded that the inference does not introduce additional rounding error if MIN operator is used. When product is used for the inference, the additional rounding error introduced might be slightly significant only if 16 bit DAC is used.

2.3.3 Defuzzyfication

Several defuzzyfication methods exists, some of them are more spread than the others [4]-[6]. One of the most popular defuzzyfication method is the Mean of Maximum (MOM), when the crisp output value is calculated as the mean of maximum values of the aggregated output fuzzy set:

$$Y = \left[\sum_{k=1}^{M} \frac{h_y^k}{M} \right], \tag{31}$$

where h_y^k denotes the heights of those points where the fuzzy set has one of its maximum value. As it can be seen, there is no additional rounding error introduced in this way. Thus, it can be concluded, that Mamdani type fuzzy controller with MIN-MAX operators and MOM defuzzyfication gives the same output as a DSP based fuzzy controller, even when 16 bit DAC is used.

Other well-known defuzzyfication methods are the Center of Area (COA) and Center of Gravity (COG) methods. The only difference between these two methods is that COA calculates the center of the aggregated fuzzy set, while COG calculates the center of the gravity of the fuzzy sets taking part in the aggregation. Thus, COG calculates twice the overlapped areas. In what follows, only the COG is presented, COA has similar reasoning. The COG defuzzyfication method is

$$Y = \left[\frac{\sum_k T^k \cdot X_G^k}{\sum_k T^k}\right] = \left[\frac{\sum_k \lambda(k)\cdot(D_x^i - A_x^i + C_x^i - B_x^i)\cdot X_G^k}{\sum_k \lambda(k)\cdot(D_x^i - A_x^i + C_x^i - B_x^i)}\right] \tag{32}$$

where X_G^k represents the center of gravity of the kth modified output membership function, while $T^k = \lambda(k)\cdot(D_x^i - A_x^i + C_x^i - B_x^i)$ is the area of this membership function. The use of an algorithm which calculates the 48 bit numerator and divides it with the 32 bit denominator is recommended in order to avoid significant rounding error. Analyzing only the division, it can be concluded that there is no additional rounding error, because anyway only the integer part is passed to the DAC. However, in the nominator the $\lambda(k)$ values are not the real ones, only the integer parts. This leads to an additional rounding error. In order to try to estimate the error introduced by the COG defuzzyfication method, let's consider P_1, P_2, Q_1, Q_2 arbitrary integer numbers, and calculate the weighted average of their quotient:

$$\frac{X_1\frac{Q_1}{P_1} + X_2\frac{Q_2}{P_2}}{\frac{Q_1}{P_1} + \frac{Q_2}{P_2}} = \frac{(N_1X_1 + N_2X_2) + X_1\frac{R_1}{P_1} + X_2\frac{R_2}{P_2}}{(N_1 + N_2) + \frac{R_1}{P_1} + \frac{R_2}{P_2}}. \tag{33}$$

It can be observed, that in relation (32) only the members of the round brackets from (33) is taken into account which leads to additional rounding error. However, it is important to notice, that omitting the terms in the numerator will cause a negative rounding error, while omitting the terms the denominator will cause a positive rounding error. Thus, their counter effect will partly extinguish the rounding error when the terms from both numerator and denominator are omitted. Because there is no simple way to analytically determine the additional rounding error caused by the COG calculated with (32), this error was determined experimentally (for a detailed result see subchapter 4.2). Experimental results show that the additional rounding error caused by COG is less than 4 bits. Here again, it can be concluded that using only 14 bit DAC the error introduced by the COG is extinguished by the DAC.

2.3.4 Sugeno Type Fuzzy Controller

Because the defuzzyfication has a significant calculation demand, another type of fuzzy controller is also used, the so called Sugeno type fuzzy controller. In this case the output of the rule is a polynomial function of the inputs, instead of a fuzzy set. The output of the controller is the weighted average of the output of the rules, where the weight is equal to the "firing" degree of the given rule. For simplicity, the most common used Sugeno type controller is the zero order one, in which case the output of the rule is a crisp value. Using MIN-MAX operators, the output of the zero order Sugeno type fuzzy controller is given as

$$Y = [y' \cdot MAXINT] = \left[\frac{\mu(x_1)\cdot x_1 + \mu(x_2)\cdot x_2}{\mu(x_1)+\mu(x_2)} \cdot MAXINT \right], \tag{34}$$

where – for simplicity – it was considered that there are only two inputs with two rules and the "firing" degree of the first rule is equal to the membership value of the first input, while the "firing" degree of the second one is equal to the membership value of the second input. The output of the first rule is equal to the first input, while the output of the second one is equal to the second input. Considering relations (9)-(11), it can be calculated the output of the zero order Sugeno type fuzzy controller as

$$Y = \left[\frac{\frac{X_1 - A_x^i}{B_x^i - A_x^i}\cdot x_1 + \frac{X_2 - A_x^j}{B_x^j - A_x^j}\cdot x_2}{\frac{X_1 - A_x^i}{B_x^i - A_x^i} + \frac{X_2 - A_x^j}{B_x^j - A_x^j}} \cdot MAXINT \right]. \tag{35}$$

After rearranging it

$$Y = \left[\frac{(X_1 - A_x^i)\cdot(B_x^j - A_x^j)\cdot X_1 + (X_2 - A_x^j)\cdot(B_x^i - A_x^i)\cdot X_2}{(X_1 - A_x^i)\cdot(B_x^j - A_x^j) + (X_2 - A_x^j)\cdot(B_x^i - A_x^i)} \right]. \tag{36}$$

The rounding error is zero when an algorithm is used which calculates the 48 bit numerator and divides it with the 32 bit denominator. A simplified and faster way is to use the MUL and DIV opcodes of the microcontroller. In order to reduce the error, both the numerator and denominator should be multiplied by $MAXINT$.

$$Y = \left[\frac{\left(\frac{(X_1 - A_x^i)\cdot MAXINT}{B_x^i - A_x^i}\cdot X_1 + \frac{(X_2 - A_x^j)\cdot MAXINT}{B_x^j - A_x^j}\cdot X_2 \right)}{\frac{(X_1 - A_x^i)\cdot MAXINT}{B_x^i - A_x^i} + \frac{(X_2 - A_x^j)\cdot MAXINT}{B_x^j - A_x^j}} \right] = \left[\frac{\mu(X_1)\cdot X_1 + \mu(X_2)\cdot X_2}{\mu(X_1)+\mu(X_2)} \right] \tag{37}$$

In conclusion, the zero order Sugeno type fuzzy controller calculated in this simple way (37) has a similar rounding error as the COG defuzzyfication method, the only difference is, that in this case a 32 bit numerator is divided by 16 bit denominator. Using only 14 bit DAC, this rounding error does not appear at the output of the DAC.

An important remark is that the presented error propagation of the fuzzy controller is true for 8 bit microcontroller as well, when 8 bit ADC and DACs are used. It can be also concluded that Mamdami type fuzzy controller with MIN-MAX operator and MOM defuzzyfication has no rounding error compared to the DSP based one. This is also true for multiplication operator and COG defuzzyfication if one uses only 14 bit DAC.

3 Implementation of the Embedded Fuzzy Controller

In this chapter an embedded fuzzy controller is presented which has been implemented in an industrial computer equipped with a 16 bit microcontroller (INTEL 80186 @ 16 MHz).

3.1 Storing the Parameters of the Membership Functions

Cheapness of an embedded system can be achieved by using low performance microcontroller connected to low capacity memory. As it was pointed out in subchapter 2.1, the parameters of the membership functions are stored as 16 bit integers and so the variables, like inputs/outputs of the controller. Thus, the memory requirement for storing the parameters and variables of the algorithm are only half or even less than a quarter of the memory capacity needed for a DSP based controller. This is due to the fact, that floating point values are usually stored in 32 bit ("single" float) or 64 bit ("double" float), sometimes even 80 bit ("extended" float) memory storage.

In the proposed fuzzy controller 3 type of membership functions can be used for the input (trapezoidal, triangle, and the generalized bell curve) and 3 type for the output (trapezoidal, triangle, and singleton). Four parameters need to be stored for the trapezoidal membership function, let's denote them with A, B, C, and D. In the same way can be stored the parameters of the triangle membership function ($B = C$). The generalized bell curve

$$f(X) = \frac{1}{1 + \left| \frac{X - B}{C} \right|^{2D}} \tag{38}$$

needs 3 parameters to be stored, where D – for the simplicity of the algorithm – only 4 values can take {0.5; 1; 1.5; 2} coded on 4 different integer values. Finally, only one value needs to be stored in B for the singleton.

For a general solution every membership function has 4 parameters, as it is shown in Table 1.

Table 1
Parameters of the membership functions

1ST PARAMETER	2ND PARAMETER	3RD PARAMETER	4TH PARAMETER
A	B	C	D

Starting from the assumption that the 1st parameter of the trapezoidal (triangle) membership function does not reach the $MAXINT$ value, just analyzing the value of the 1st parameter , it can be decided the type of the membership function stored in a memory location (If the 1st parameter is equal to $MAXINT$ a singleton is defined, and can be identified with only one parameter). If parameter A is less than $MAXINT$, a trapezoidal (triangle) membership function, otherwise a generalized bell curve for the input, or singleton for the output is stored. The number of the input and output membership function can be arbitrary large, however, for the proposed fuzzy controller it has been limited to 4, respective to 2. In the same way, the number of membership functions for an input/output can be arbitrary large, but in the proposed fuzzy controller it has been limited to 8. Although, the number of membership functions for an input/output is generally even number, 8 is a power of two, which helps omitting multiplication when accessing the memory location. Thus, the memory content, which stores the k th parameter of j th membership function of i th input/output, is addressed in the following way (C programming style notation):

$$T[i][j][k] = T_{addr} + i << 6 + j << 3 + k << 2 + H/L, \qquad (39)$$

where $<<$ denotes the left shift operator (which is equivalent with multiplying by the power of 2), and H/L is set to zero for the lower byte, respective to one for the higher byte of the integer value. Important remark is, that all indices start from zero! T_{addr} represents the absolute value of the first byte of the table, the rest represents the offset of a parameter. The memory is organized in byte form (the width of the data bus is one byte), and the memory capacity needed to store al the parameters is $8 \cdot 8 \cdot 6 = 384$ bytes. Additional 6 bytes are needed to store the number of the membership functions used for each input/output (if this value is set to zero, the given input/output is not used!).

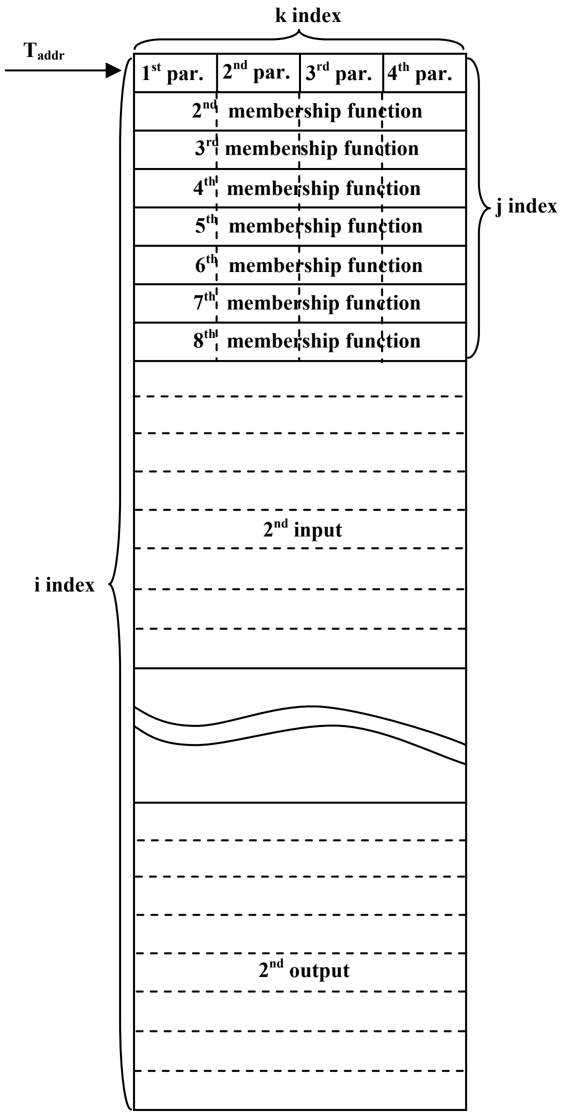


Figure 3
The parameters of the membership functions stored in memory

3.2 Constructing the Algorithm

In this chapter some useful hints are presented to construct the algorithm. See [3] for a simplified pseudo code of the implemented algorithm.

3.2.1 Fuzzyfication

For each input/output 2 bytes of memory location are used where the current input/output value of the fuzzy controller can be stored. This needs additional 12 bytes of memory location. The fuzzyfication function used in the implementation is the identity function, e.g. *f(y)=y*. That means, that input variables are not fuzzyfied, instead they are employed in the inference process directly. Although, the fuzzyfication process means only the fuzzyfication of each input variable, in what follows the calculation of the degree of consistency between the input value and the membership functions of the appropriate input is also included.

In case of trapezoidal (triangle) membership function the degree of consistency is calculated as in (14), which needs one multiplication and one division. In order to omit the multiplication, the following trick is used:

$$MAXINT \cdot X = FFFF_H \cdot X = (10000_H - 1) \cdot X = (X << 16) - X \,, \tag{40}$$

thus, subtraction is used instead of multiplication. In the worst case only a division is needed, otherwise the degree of consistency is either 0 or 1. In case of generalized bell curve the precision of the division can be increased if both the numerator and denominator are multiplied by 256 (shifted with one byte to the left). For example, when $D = 0.5$ the following relation gives an approximate value:

$$\mu(X) = \frac{FFFF00_H}{100_H + \dfrac{|X - B| << 8}{C}} \,. \tag{41}$$

Storing the $\mu(X)$ membership value needs two bytes, and current membership values are stored in similar way as the parameters of the membership functions.

3.2.2 Inference

Compactness and effectiveness has taken with first priority when the codification of the rules has been implemented. Each rule consists of two parts: one antecedent part containing one or more antecedent term, and one consequent part having one consequent term. Each antecedent/consequent term needs one byte for storage as it can be seen in Figure 4. The end of the rule-base is indicated by a value greater than 128 (the most significant bit is set to 1).

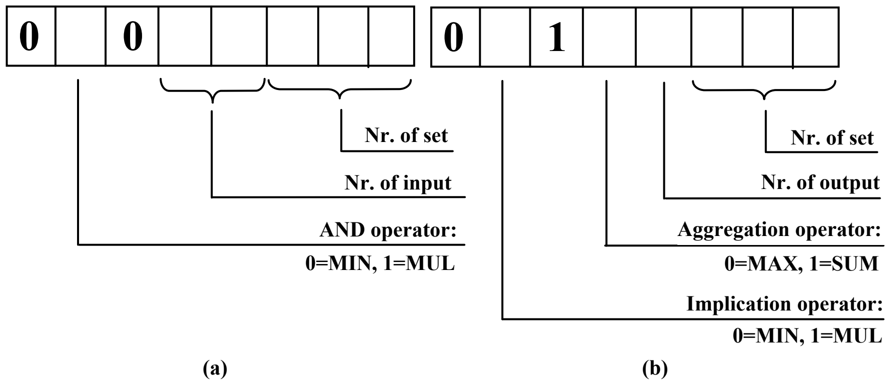


Figure 4
Coding the antecedent (a) and consequent (b) term of the rule

It should be noted that in the antecedent term the number of the membership function (set) and the number of input is stored in such way, that the memory location of the parameter can be calculated by masking the antecedent term, then shifting to the left by 3 and adding the number of the parameter, as it is in (39). In the same manner is coded the consequent term. In the antecedent term the AND operator (MIN - minimum or MUL - multiplication) is also coded. In the consequent term the implication operator (MIN - minimum or MUL - multiplication) and the aggregation operator (MAX - maximum or SUM - bounded sum) are coded. The number of bytes required for storing a single rule is maximum 5, since to each input one antecedent term corresponds, while the rule base requires in the worst case 20480 bytes (counted for 4096 rules). However, in everyday applications the number of rules is limited to around 100.

3.2.3 Defuzzyfication

When singletons are defined at the output (Sugeno type fuzzy controller) the defuzzyfication is simple and there is no need to describe in details. In case of COG it has been used the trick, that the division can be avoided if the divider is equal to $MAXINT$. This is due to the fact, that having a 32 bit dividend and 16 bit divider, the latter equal to $MAXINT$, the result is the high word of the dividend.

4 Results

4.1 Simulation Results

Most fuzzy engines are analyzed for three basic parameters: performance, code size, and inference time. Performance involves the smoothness of output in transition areas (i.e. where the input membership functions overlap). Performance was examined by building two fuzzy controllers in a MATLAB environment, one using floating point calculation, and the other one using only integer values. Both fuzzy controllers have 2 inputs and one output, having the following rule base:

1. (In1==NB) & (In2==NB) => (Out=NB)
2. (In1==NB) & (In2==ZR) => (Out=NB)
3. (In1==NB) & (In2==PB) => (Out=ZR)
4. (In1==ZR) & (In2==NB) => (Out=NB)
5. (In1==ZR) & (In2==ZR) => (Out=ZR)
6. (In1==ZR) & (In2==PB) => (Out=PB)
7. (In1==PB) & (In2==NB) => (Out=ZR)
8. (In1==PB) & (In2==ZR) => (Out=PB)
9. (In1==PB) & (In2==PB) => (Out=PB)

The obtained surfaces can be seen in Figure 5-8. In Figure 5 Mamdani type fuzzy controller is presented with MIN-MAX operator, trapezoidal membership function, and COG defuzzyfication. There is no visible difference between the DSP and μC based fuzzy controller, the average difference is only 1-2 bit when using 16 bit DAC. In Figure 6 Mamdani type fuzzy controller is presented with MUL-SUM operators, trapezoidal membership function, and COG defuzzyfication. It can be observed that the surface of the μC based fuzzy controller around zero is flatter than the surface of the DSP based controller when 16 bit DAC is used. This difference is due to the presence of the rounding error of the μC based fuzzy controller.

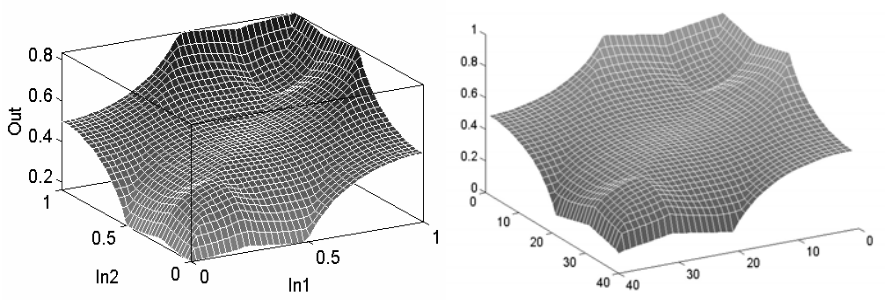


Figure 5
Surface of the Mamdani type fuzzy controller with MIN-MAX operators, trapezoidal membership functions, and COG defuzzyfication (DSP based on the left, μC based on the right)

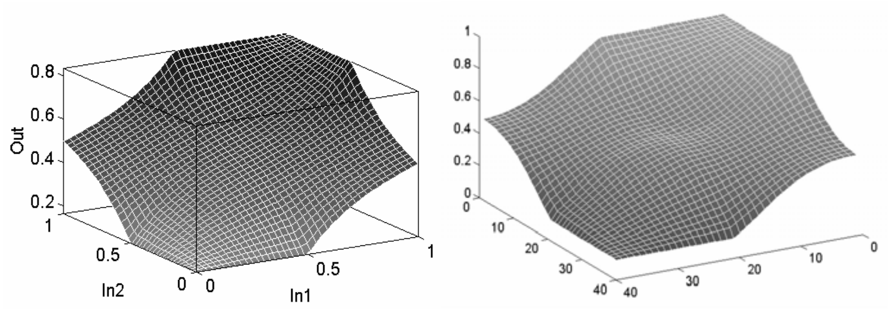


Figure 6
Surface of the Mamdani type fuzzy controller with MUL-SUM operators, trapezoidal membership functions, and COG defuzzyfication (DSP based on the left, μC based on the right)

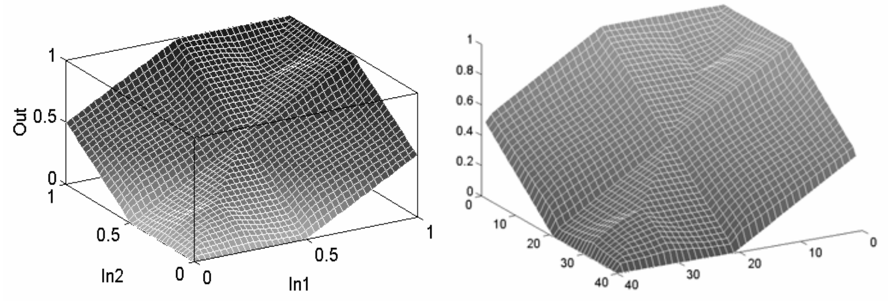


Figure 7
Surface of the Sugeno type fuzzy controller with MIN-MAX operators, and trapezoidal membership functions (DSP based on the left, μC based on the right)

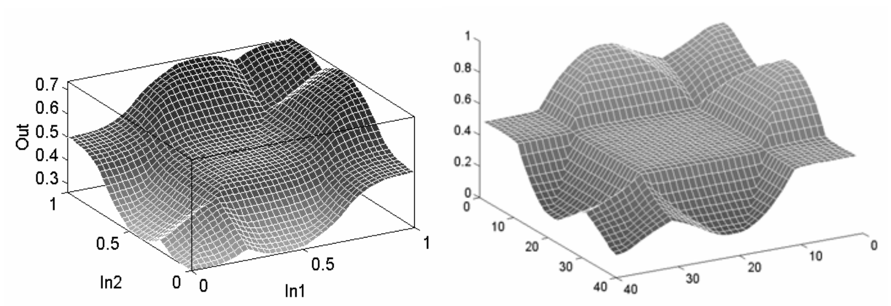


Figure 8
Surface of the Mamdani type fuzzy controller with MIN-MAX operators, generalized bell curve membership functions, and COG defuzzyfication (DSP based on the left, μC based on the right).

In Figure 7 the surfaces of the zero order Sugeno type fuzzy controller with MIN-MAX operators, and trapezoidal membership functions are presented. As it can be seen there is no visible difference between the DSP and μC based fuzzy controller, the average difference is only 1-2 bit when using 16 bit DAC. Finally, in Figure 8 the Mamdani type fuzzy controller with MIN-MAX operators, generalized bell curve membership functions and COG defuzzyfication is shown. In this case, the rounding error is significant, and is presented only for the sake of completeness, its usefulness might be questionable for some applications.

4.2 Experimental Results

The proposed embedded fuzzy controller was implemented in an industrial computer (UNI-PLC-100) at Process Control Ltd. This industrial computer is based on INTEL 80186 microcontroller at 16 MHz, and is equipped with analogue and digital input/output boards. The front panel of the industrial computer is presented in Figure 9.

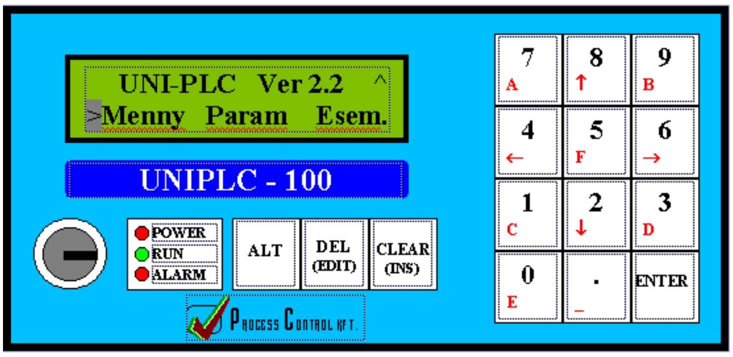


Figure 9
The front panel of the industrial computer. Courtesy of the Process Control Ltd.

There are two possibilities to store the parameters of the membership functions and the rule base: either entering manually with the help of the keyboard on the front panel, or downloading through the serial line from a Personal Computer (PC).

Because the whole fuzzy controller algorithm is implemented in assembler language, the code size is very compact. The overall code is less than 10 KB. The inference time is also impressive, taking into account that the microcontroller runs only at 16 MHz. Some inference time for Mamdani and Sugeno type fuzzy controller with MIN-MAX operators, trapezoidal membership functions and COG defuzzyfication (only for the Mamdani) are presented in Table 2. It can be seen that the total inference time (fuzzyfication+inference+defuzzyfication) is very impressive for a fuzzy controller with 2 inputs, 3 membership functions (sets) per input, 9 rules and COG defuzzyfication. It is only 1011 μs, which means that the fuzzy controller is able almost every 1 ms to update the control signal.

Even though a DSP based fuzzy controller might be able to update the control signal 100 times more frequently than the microcontroller based one, this is useless if the controlled process is slow, and its state variables changes so slowly, that there is no effect on the controlled process if one update the control signal more frequently.

Table 2
Inference time with MIN-MAX operators

Nr. of input	Nr. of set/input	Nr. of rules	Fuzzification (trapezoidal) [μs]	Inference [μs]	Defuzzyfication Sugeno / COG [μs]
1	3	3	78	50	85 / 641
1	5	5	96	70	94 / 657
1	7	7	115	90	100 / 660
2	3	9	132	169	95 / 710
2	5	25	177	427	96 / 820
2	7	49	222	819	122 / 1070
3	3	27	168	593	127 / 1123

Some inference time for Mamdani and Sugeno type fuzzy controller with MUL-SUM operators, trapezoidal membership functions and COG defuzzyfication (only for the Mamdani) are presented in Table 3. The same remark can be mentioned here, the inference time is small compared to the clock frequency of the microcontroller.

Table 3
Inference time with MUL-SUM operators

Nr. of input	Nr. of set/input	Nr. of rules	Fuzzification (trapezoidal) [μs]	Inference [μs]	Defuzzyfication Sugeno / COG [μs]
1	3	3	78	53	85 / 910
1	5	5	96	74	94 / 918
1	7	7	115	96	100 / 922
2	3	9	132	197	100 / 1056
2	5	25	177	505	101 / 1348
2	7	49	222	971	122 / 1903
3	3	27	168	736	127 / 1900

Conclusions

In this article the implementation of an embedded fuzzy controller is proposed for 16 bit microcontroller with fast fuzzyfication-inference-defuzzyfication algorithm. It has been demonstrated that because controllers receive information from the process via ADCs and controls the process with the help of DACs the use of DSP based fuzzy controller might not pay the effort for some type of fuzzy controller.

Generally, 16 or only 14 bit DACs are used at the output of a fuzzy controller, in which case the precision of the floating point operation is lost and a DSP based fuzzy controller seems better, faster and more accurate than it is in reality, especially when the state variables of the controlled process change slowly. Thus, the use of a μC based fuzzy controller with integer operations is a reasonable compromise between performance and price.

Simulation and experimental results has been also presented which are supporting the theoretical idea presented in this article.

Acknowledgement

This paper was supported by the Hungarian N.Sc. Fund (OTKA No. T 042866) for which the authors express their sincere gratitude.

References

[1] Ferenc Farkas, Szilárd Varga, Aleksei Zakharov: Investigation of DC servo drives with fuzzy logic control, Czasopismo Techniczne 4E/1998, Wydawnictwo Politechniki Krakowskiej, Poland, 1998, pp. 35-45

[2] Ferenc Farkas, Aleksei Zakharov, Szilárd Varga: Speed and position controller for DC drives using fuzzy logic, Studies in Applied Electromagnetics and Mechanics (Vol. 16), Applied Electromagnetics and Computational technology II, Amsterdam, Netherlands, IOS Press, 2000, pp. 213-220

[3] Ferenc Farkas: Implementation of fuzzy controller on 16 bit microcontroller, Proceedings of IEEE International Conference on Intelligent Engineering System (INES'99), Stara Lesna, Slovakia, 1-3 November 1999, pp. 567-572

[4] George J. Klir, Bo Yuan: Fuzzy sets and fuzzy logic - Theory and applications, Prentice Hall, New Jersey, 1995

[5] Li-Xin Wang: Adaptive fuzzy systems and control - Design and stability analysis, Prentice Hall, New Jersey, 1994

[6] Retter Gyula: Fuzzy, neurális, genetikus, kaotikus rendszerek – Bevezetés a „lágy számítás” módszereibe, Invest Marketing Bt., Budapest, 2003

[7] Kai Michels: Fuzzy control of electric drive?, European Conference on Power Electronics (EPE'97), Trondheim, Norway, 8-10 Sept, 1997, pp. 1.102-1.106

Coping with Security in Programming

Frank Schindler

Department of Applied Computer Science and Engineering
Faculty of Electrical Engineering and Information Technology
Slovak Technical University, Iľkovičová 3, SK-812 19 Bratislava, Slovakia
e-mail: frank.schindler@stuba.sk

Abstract: This article deals with importance of security issues in computer programming. Secure software can only be designed with security as a primary goal. To achieve that we would have to redesign our computer systems with security in our mind including entire computer environment, e.g. hardware, programming languages and, of course, operating systems. In software development process the quality of resulting computer code should be the most important aspect during the whole program development process. Simplicity of the code in computer programs always pays off. Extra options and features can result in unmanageable complexity. For computer security purposes, program modularisation is of a paramount importance and seems to be the only way how to properly cope with complexity. Internal consistency of the whole program should be frequently checked via assertions. They are the best way to check parameter validity coming from other program units e.g. modules. Especially each module must distrust everything else coming from other modules and/or from the user. Frequently used code optimisations are classically leading to some sort of redundant code options and features and thus indirectly causing a useless code complexity. Extensive testing of programs is necessary for finding possible bugs in programs. However it does not locate security holes in the system. Standard software implementation techniques are completely inadequate in the production of a secure code. Consequently an introductory programming course as a college course should be taught in parallel with introductory security of computer systems, since it is too late to teach it as an elective at the end of computer science curriculum. In general, security of computer systems and programming should not be separated as two different and separate disciplines instead of it they should be integrated together.

Keywords: security of computer systems, principle of least privileges, lack of functionality, module, module's specification, module's implementation

1 Introduction

Standard software development techniques are completely inadequate to create secure software, since they only deal with correctness of software e.g. with its specified functionality. If you press the key A, then action B will happen.

Consequently a correct program behaves exactly according to its specification. On the other hand, secure software relates to a lack of functionality (see [3]). No matter what the user (attacker) does, he cannot do action X. It is possible to test the functionality of a program, but there is no known way to test the lack of it.

In real life situations, there are many different ways a computer program can be made to fail or crash. Often this may be easily achieved when the user (attacker) provides invalid input either on purpose or by an accident. Deliberate actions on the side of the user could also include feeding the program with viscous data. Programs react on such inputs in various ways. Some of them simply fail without any error messages, others act incorrectly, and yet others crash the whole operating system. A program that crashes the whole system is unacceptable above all in the computer security area, because it may be possible leading to some sort of security breach. Therefore development of secure software is sufficiently different from programming other software applications. Common program's bugs (e.g. buffer overflows) are the most serious security problems in today’s computer systems. To be more specific the biggest problem in computer security is related to the weakest link property. In old days of computing a programmer received a task to be performed, went away and developed the whole program alone. Nowadays programs for complex tasks are programmed by a team of many different programmers that can produce millions of lines of code. The level of output of such a typical programmer involved in the team is on average only 5 or 10 lines of code per day. Large programs require precise design documents showing what each piece of code does and how it interacts with the rest of the program. Bug-free code is duty for all developers on the team, therefore they have to be involved in peer design reviews and peer code reviews. When a programmer finishes a particular part of code other team members must make a complete walk-through of the programmer's design and/or code. Basic principles of software engineering advocate to write small, self-contained program units called modules. Each module should be isolated from harmful effects of other modules. This can be achieved via information hiding (encapsulation). Secure programming implies that each program's (or subprogram's) actions must be contained in the program's specification. To write safe and sound programs we should stick to three basic principles: information hiding, defensive (robust) programming, and assuming the impossible.

2 Information Hiding (Encapsulation)

A module often specifies and implements an abstraction (see [4]). The module’s specification describes the behavior and properties of the abstraction, and the module’s implementation contains the concrete realization in the program code. Effective programming also takes advantage of reusing existing code libraries and/or off-the-shelf (reusable) components. Each well-designed module should encapsulate (group and hide) private/public data and the code bodies. Moreover the module should provide well-defined interfaces through which the program can access or modify module's data. Any undocumented, unspecified code options, side effects or function definitions may result in a covert channel or trap door (back door) through which data could either leak, or be changed or damaged and thus they could pose a severe security problem. Most of these ideas are a direct consequence of the "Principle of Least Privilege" and they form the basis for program code integrity and security. During the whole process of the program specification, design, implementation, testing and maintenance keep in mind that simplicity always pays off.

3 Defensive Programming

It is based on the idea that the given program, being executed, should not depend on anything that is not self-created in the program. Every time when a user (attacker) is running the program the programmer must suppose that the user may break it either intentionally or unintentionally by providing flawed input to it. Therefore, insert into your code as many of assertions as possible to catch erroneous data flowing from function (module) to the other function (module). By no means: "garbage-in garbage out". That would be fatal. On the other hand the programmer should never abuse the features of the given programming language like pointers. Especially de-allocation of pointers is a very dangerous operation often leading to dangling pointers. On the other hand, if a pointer is returned by a function it allows illegal access to program's data (data leakage vulnerability). Also the same holds for array indices. Another potential weakness may relate to error codes returned from the functions. When they are left unchecked the values returned from the function should not be used, because they could act as a destructive trash when they are used as input for rest of the code. Avoid use of cryptic code or extra features and options of the programming language that only very few programmers know and use. Technique of defensive programming is often referred to as "robust programming" (see [1]).

4 Assuming the Impossible

Module's specification should be used as a document against which the program should be tested after it has been finished. Without it there is no rigorous way to describe what has been accomplished in the program. Consequently it has to be complete and up-to-date. Let me rephrase it. Anything that is not in there does not have to be implemented in the code. Programmers first write a program and then they test it to see if it functions correctly. Then, if any bugs are found they fix them and try it again. This process cannot lead to a completely correct program, since this way we are unable to show absence of bugs. Such a program usually works fine in the most typical situations. To verify that a program is correct is way over our capabilities today. Nevertheless we can try to do our best when we are testing programs. Generally, two types of tests are needed. The first one should be generic one made from the module's specification. The second set of tests should examine module's implementation limits, e.g. buffer management errors. Perhaps, designing, writing and testing of a secure computer program could be best compared with driving the car on the busy highway. Programming language, we are using, should never be misused the same way as the car. Defensive driving is a counterpart of defensive programming in this case. Some features like pointers should be used with caution the same way as when we are riding the car we must anticipate that anytime there could be a cat or some other animal running from behind the bush into our way. Therefore numerous assertions should be embedded in the source code in order to improve the quality of code. Anytime in the program there is a possibilities to check the internal consistency of the system you should include an assertion. If it fails it can abort the program and report what was going wrong. This way it is possible to catch up a lot of errors from which some of them could lead to a serious security breach. Producing wrong answers in the program can do a lot more harm. Do not allow garbage data to propagate freely through your program! In order to illustrate some basic concepts, here I provide a few short and simple examples concerning secure programming. Remember that most computing errors happen exactly in cases like these.

5 Programming Examples

5.1 Buffer Overflows

```
int i = 0;
int a[10];          // Here, buffer is allocated as an array made of 10 integers
....
i = 11;             // index i is set over the upper bound of the array
a[i] = 1;
....
// Buffer overflows cause about 50% of the security problems on the
// Internet (Ferguson, 2003).  Algol 60 solved this problem!
// However C, C++ allow buffer overflows!
// Solution: Avoid any such language for secure applications!
```

5.2 Missing Initialization of Variables

```
#include <stddef.h>       //                   for NULL
...
int    *ptr;              //                   int      *ptr = NULL;
...
free(ptr);
```

// here a pointer is declared, that has not been allocated, but it is de-allocated. // This leads to a typical situation in which operating system can crash.

5.3 Cryptic Code

```
char *p1, *p2;
...
while (*p1++ = *p2++)
          ;
// *p1++ is equivalent to:
// *(p1++)  it is a unary operator...right to left
```

5.4 Program Ignoring Error Messages

```
const int NO_ERROR = 0;
....
int  One_Function(char ch, char *ptr);
int err_Message;
char u, v;
....
err_Message = One_Function('?',&v);
u = v;
// it should be:  if  (err_Message == NO_ERROR)  u = v;
```

5.5 Missing Null-Condition Restrictions

```
int  a[5];                    // stack is represented as an array made of 5 integers
int  top = -1;                // this condition means the stack is empty
...
```

```
i = pop();                    // trying to pop the top element from an empty stack
// it always must  be:
if (top != -1)                // see if the stack is non-empty and then pop
   i = pop();
```

5.6 Dangling Pointers

```
#include <stddef.h>       // for NULL
...
int    *ptr1 = new int;
int    *ptr2 = new int;
*ptr2 = 42;
*ptr1  = *ptr2;
delete ptr1;              // avoid an inaccessible object
ptr1 = ptr2;
delete ptr2;
// Notice a missing assignment:
ptr1 = NULL;               // avoid dangling pointer
```

5.7 Assertions

```
#include <assert.h>
// C++ standard library providing executable assertions
.....
double FindAverage ( int    sumOfScores, int    studentCount )
// Precondition:
//                  sumOfScores is set
//                  studentCount > 0
// Postcondition:
//                  average = sumOfScores / studentCount
{
      double average = 0.0;

      assert (studentCount > 0);
```

```
    // this function halts the program if the expression is false
    average = sumOfScores / studentCount;
    return average;
}
```

5.8 Work with Library Functions

It seems to make a perfect sense to ask students to write a program calling some of the standard input/output library functions in C. Their task is to feed them with such an input, that will crash the whole operating system. When that happens they should use debugger to see why the system collapsed.

Example:

```
char a, b, c, s[10];
int   n;
double  x;
scanf ("%c%c%c%d%s%lf", &a, &b, &c, &n, s, &x);
```

As a direct continuation of this assignment students should be asked to write a robust scanf function in C of the same type.

Examples of this sort could be quite useful during the lab sessions associated with the corresponding face-to-face lectures!

5.9 Paradoxes – Testing the Impossible (Variant of "Y2K Bug")

An insurance company offers a life insurance to its clients based on their age. It uses a program reading from a file a list of people consisting of the name and year of birth per line. The year of birth is a two-digit item. Assume we are in 20th century and today we have the first day of January 2000.

Age of the person born in 1925 can be obtained as:

"00" - "25" = -25 years.

Conclusion

Software manufacturers are used to sell their products with many known bugs and therefore they disclaim any legal liability for using their merchandise in the corresponding software license. For some applications like computer games this is not a real problem. However computer programs are more and more used everywhere in our lives. Standard implementation techniques are completely inadequate to create robust software. Software security can be guaranteed only if all program parts do their job. Low-quality code is the most common cause of real world attacks, and should be avoided. A secure program may just be written by the sound technique of defensive (robust) programming coupled along with information hiding provided it has been tested thoroughly. The mission to test large programs with millions of lines of code is almost impossible. Only thousands (or millions) of users might test it exhaustively. In college programming courses the special attention should be paid to the secure design, implementation and testing of programs systematically. It is not enough when a program runs satisfactorily for correct data only. Moreover secure programs should not cause any unnecessary vulnerability such as information damage, leakage or data diddling by strictly adhering to "Principle of Least Privilege" and information hiding. Although this approach is not going to solve all security problems lurking on us in our programs it could lead to programs that are better structured, better tested, better thought of and at last more secure. The only imaginable way to make secure software would be to redesign our entire computer environment, including hardware, programming languages and operating systems with security as a primary goal in our mind. And, that is time consuming and costly.

References

[1] M. Bishop, D. Frincke: Teaching Robust Programming, IEEE Security and Privacy, published by IEEE Computer Society, Vol. 2, No. 2, (2004) pp. 54-57

[2] M. Blaha: A Copper Bullet for Software Quality Management, Computer published by IEEE Computer Society, Vol. 37, No. 2, (2004) pp. 21-25

[3] N. Ferguson, B. Schneier: Practical Cryptography, Wiley Publishing Inc. Indianapolis, Indiana, (2003) ISBN 0-471-22357-3

[4] M. R. Headington, D. D. Riley: Data Abstraction and Structures with C++, D. C. Heath and Company, Lexington, MA, (1994) ISBN 0-669-29220-6

[5] N. R. Mead: Who is Liable for Insecure Systems? COMPUTER published by IEEE Computer Society, Vol. 37, No. 7, (2004) pp. 27-34

[6] S. Meyers: Effective C++: 50 Specific Ways to Improve Your Programs and Designs, Addison-Wesley, Reading, Massachusetts (1997)

[7] C. P. Pfleger: Security in Computing, Prentice-Hall International, Inc., Upper Saddle River, NJ, (1997) ISBN 0-13-185794-0

[8] F. Schindler: Coping with Safe Programming, Proceedings of Conference "I & IT 2004", Banská Bystrica, Slovakia, (2004) pp. 142-145, ISBN 80-8033-017-7

The Statics of the Traditional Hungarian Composite Reflex Bow

Sándor Horváth, Géza Körtvélyesi, László Legeza

Bánki Donát Faculty of Mechanical Engineering, Budapest Tech
Népszínház u. 8, H-1081 Budapest, Hungary
horvath.sandor@bgk.bmf.hu

Abstract: The operation of the Hungarian bow raises several fascinating mechanical questions. To answer these questions a good number of experiments and calculations need to be made, moreover the mechanical model of the bow is needed to be prepared which appropriately confirm the results of experiments. Teachers in the Bánki Donát Mechanical Engineering College of Budapest Polytechnic set up a small laboratory in 1997 in order to study and measure the physical characteristics of traditional bows. The mechanical analysis of bows is based on the experiments gained in the laboratory and the results of measurements. The knowledge acquired about the mechanical model of bows facilitates not only the analysis of the traditional Hungarian bow, but also provides a good foundation for the comparison from the technical point of view of various composite reflex bows belonging to different historic ethnic groups.

1 Introduction

In the course of the history of mankind certain peoples and nationalities can always be traced to have risen and fallen and it is primarily the historians' task to research in the circumstances. According to historians, in many cases the immediate reason for certain peoples' rise was the ability to set up the best-organised and most disciplined army of their age, which was equipped with the most advanced weaponry.

In the history of Hungarians there was a period of at least one and a half centuries in which Hungarians had by far the most powerful army of their time. There is written evidence proving that princes or pretenders in western countries often requested Hungarians still living in Etelköz (i.e. the homeland of nomadic Hungarians in Asia) to support them with their tribes. Hungarians had a good reputation worldwide for their modern, well-organised and well-disciplined warfare, which bore a lot of resemblance to the Huns' army. Their most efficient weapon, the composite reflex bow, which was exclusively used by eastern nomadic tribes, was regarded as crucially decisive for every battle. After the

Hungarian tribes had occupied their new homeland in the Carpathian Basin, almost the whole of Europe paid taxes to the Hungarian principality' in return for the support of their invincible army. The taxes Hungarians imposed on western civilisations assured peace and quiet for them, on the other hand if they had failed to pay their taxes, the "roaming" Hungarian tribes soon appeared on the horizon claiming their share. Legend has it that the inhabitants of medieval Modena had been found in their church praying to God in the following manner: "...Almighty God, please save us from the arrows of Hungarians."

The ancient weapon called the reflex bow had been widely used for hunting and fighting by nomadic tribes in the steppes. While preserving its basic operational principal, the different tribes produced their own versions of the original weapon by developing new geometrical varieties. As a result, the Hungarian bow can be distinguished fairly easily from the Hun, Avarian, Mongolian, Chinese or Turkish bows.

In Hungary, the ethnographer Károly Cs. Sebestyén was the first who had identified the remains of the ancient Hungarian bow with the long flat bone blades which were similar to knives and had been found arranged in similar patterns in some of the graves from the time of the Hungarian Conquest of the Carpathian Basin. They have obviously meant an almost indecipherable riddle for archeologists. The bone blades covered and decorated the grip areas and their rigid ends, the horns of bows. Károly Cs. Sebestyén's articles had focused attention to the Hungarians' ancient weapon. It was Kálmán Jakus, a Physical Education teacher at Lónyai Street Reformist Academic Grammar School, though, who succeeded in manufacturing the first Hungarian bow. His primary purpose was to develop an efficient bow for sport. One of the most prominent of the next generation of developers was Dr. Gyula Fábián (1915-1985), a department head at the University of Agriculture in Gödöllő (present day Szent István University) who had carried out scientific research into the evolution of the Hungarian bow, moreover he had also been able to make a reconstruction of the traditional reflex bow. His reconstructions were also acknowledged by archeologists specialising in the given historic period. His attempts have been followed by more or less successful reconstructions of bows. In the past few decades new bows have appeared with some metal or fibreglass parts in their construction. They also contain some plastic, and therefore proved to be much stronger than the traditional constructions. Manufacturing bows which are exclusively made of natural materials is more time-consuming, requires more expertise, moreover the acquisition of special raw materials such as animal sinew, ox horn, special glue or resin etc. would make the whole process extremely difficult. Although the so-called "Hungarian Conquest period" bows available for sale these days are based on the functional and geometrical construction of their traditional Hungarian counterparts, it must be noted that their flexible bow arms are made of plastic containing glass fibre or carbon fibre.

Among Professor Gyula Fábián's disciples, Imre Puskás and Csaba Búza were the most outstanding. Árpád Ambrózy also needs to be remembered since he wrote a book about hunting archery in 1994. In this field Gábor Szőllősy should also be referred to as he was the first in Hungary to have done his doctorate in archery, moreover he has written several scientific articles and given a great number of lectures to express appreciation for the traditional Hungarian bow which is regarded as a significant product of ancient Hungarian craftsmanship as well as a brilliant "technological" achievement. He also puts great emphasis on the balanced relationship between the forces in humans and the bow. Today several manufacturers specialise in manufacturing the Hungarian bow, nevertheless Lajos Kassai's and Csaba Grózer's bows are by far the most popular.

The operation of the Hungarian bow along with the special backward shooting technique, which was so much favoured by our ancestors, raises several fascinating mechanical questions. To answer these questions a good number of experiments and calculations need to be made, moreover the mechanical model of the bow is needed to be prepared which may appropriately confirm the results of experiments. Teachers in the Bánki Donát Mechanical Engineering College of Budapest Polytechnic with the professional assistance of Dr. Gábor Szőllősy set up a small laboratory in 1997 in order to study and measure the physical characteristics of traditional Hungarian bows. The mechanical analysis of bows is based on the experiments gained in the laboratory and the results of measurements.

Our objective was to prepare the mechanical model of the Hungarian bow and then the preparation of a computer program which can be used for examinations about the geometrical optimalisation of the bow from the energetic point of view. The knowledge acquired about the mechanical model of bows facilitates not only the analysis of the traditional Hungarian bow, but also provides a good foundation for the comparison from the technical point of view of various composite reflex bows belonging to different historic ethnic groups.

2 The *Mechanical Model* of the Bow

Before starting any mechanical calculations, the geometrical features, more precisely the identifiable characteristic points of the bow need to be unambiguously defined, thus defining their position and co-ordinates with the minimum of measuring errors. It is a good idea to begin with the situation of the characteristic points and examine how they are related to each other. The measurements are related to each other as they form a measurement chain, therefore it makes checking easier. Minor mistakes might be made, though, if distances are measured instead of the characteristic angles, and then the figures calculated and concluded from the distances of angles are compared with each other.

The geometry of the drawn Hungarian bow is calculated and concluded as in Figure 1.

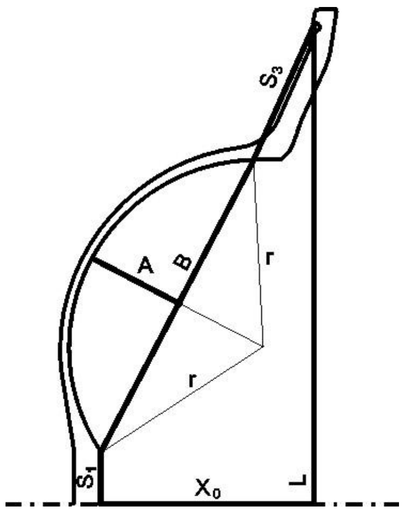


Figure 1
Geometry of the drawn Hungarian bow

in which

$2s_1$	– the length of the grip section (rigid part in the middle) (mm)
s_3	– the length of the axis of the rigid horn (mm)
$2L$	– the length of the string (mm)
A	– the biggest distance between the flexible bow (bow arm) and its string (mm)
B	– the length of the geometrical string of the flexible bow (mm)
x_0	– the distance between the string and the grip section, the height of the drawing of the bow (mm)

In order to make the calculations simpler, the following assumptions can be made:

- the bow arm forms a curved line,
- the bow is perfectly symmetrical,
- the cross section of the bow arm is constant,
- the connection between the bow arms and the rigid parts of the bow is like bracketing,

- the material of the bow arm is homogeneous and flexible,
- the grip and the horn are rigid,
- the effect of the pre-stretching of the bow is fully contained in the characteristics of the bow.

Later, after the mechanical model has been necessarily adjusted and made more precise, the assumptions above can be ignored.

The most important geometrical characteristics of the drawn condition of the bow are shown in Figure 2.

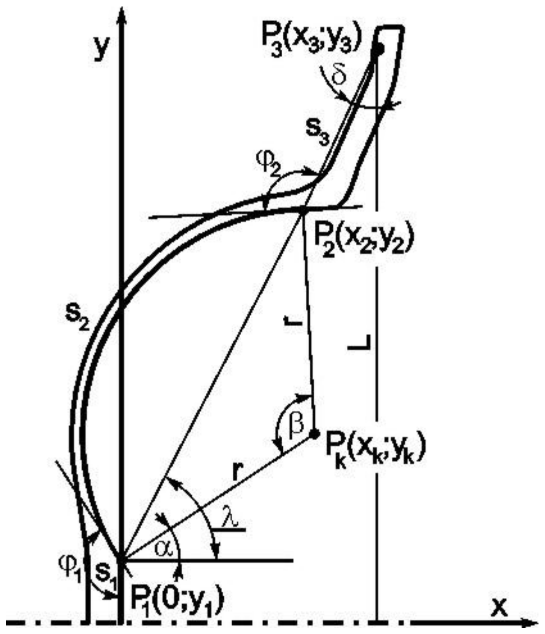


Figure 2
Geometrical characteristics of the drawn condition of the bow

The calculation of the radius of the flexible curved line:

$$r = \frac{A}{2} + \frac{B^2}{8A}, \tag{1}$$

then the equation about the central angle of the curved line:

$$\sin\frac{\beta}{2} = \frac{4AB}{4A^2 + B^2}. \tag{2}$$

With the knowledge of the results of the equations above, the length of the flexible curved line can be calculated as follows:

$$s_2 = r\beta . \tag{3}$$

Considering basic geometry, the horizontal projection of the lenght of the rigid horn (s_3) can be calculated as follows:

$$x_{S3} = \frac{ab - \sqrt{a^2b^2 - (1+b^2)\cdot(a^2 - s_3^2)}}{1+b^2}, \tag{4}$$

in which $a = \dfrac{x_0^2 + (L-s_1)^2 + s_3^2 - B^2}{2\cdot(L-s_1)}$, (5)

and $b = \dfrac{x_0}{L-s_1}$. (6)

With the knowledge of x_{S3}, the λ angle between the string of the flexible bow and the horizontal x axis can be concluded as follows:

$$\cos\lambda = \frac{x_0 - x_{S3}}{B} . \tag{7}$$

This way the α angle is calculated:

$$\alpha = \frac{\beta}{2} + \lambda - \frac{\pi}{2} . \tag{8}$$

The flexible bow fixed to the grip section shall be regarded as rigid, therefore the φ_1 angle between them is considered constant. Therefore its calculation:

$$\varphi_1 = \frac{3}{2}\pi - \frac{\beta}{2} - \lambda . \tag{9}$$

The coordinates of the characteristic P_1, P_2, P_3 and P_k points as indicated in Figure 2.

$$x_1 = 0 ; \qquad y_1 = s_1 , \tag{10}$$

$$x_2 = B\cdot\cos\lambda ; \qquad y_2 = s_1 + B\cdot\sin\lambda , \tag{11}$$

$$x_3 = x_2 + s_3\cdot\sin\delta ; \qquad y_3 = L , \tag{12}$$

$$x_k = r\cdot\cos\alpha ; \qquad y_k = s_1 + r\cdot\sin\alpha , \tag{13}$$

in which the δ angle between the rigid horn and string can be calculated from the angle function below as follows:

$$\cos\delta = \frac{L - y_2}{s_3} \,. \tag{14}$$

Finally, the calculation of the φ_2 angle, which is characteristic of the rigid context of the flexible curved line and the rigid horn, therefore can be regarded as a constant figure:

$$\varphi_2 = \alpha - \beta + \delta + \pi \,. \tag{15}$$

In order to check the geometrical figures, it is recommended to also check the figures of the bow with measuring (by measuring distance and angle), and to draw and construct a picture of the bow. The mechanical calculations about the bow can only be made if correct geometrical characteristics are available.

3 The Statics of the Drawn Bow

The basic static figures of the mechanical calculations of the traditional Hungarian composite reflex bows is the product [Nmm^2] of multiplying the F_x drawing force [N], the flexibility modulus of the material of the bow (I) and the secondary momentum of the cross-section of the bow (E).

In order to minimize the errors in the calculation due to the above-mentioned assumptions, a correctional function needs to be applied which modifies the IE product of multificaton according to the size of the deformation and to what extent the bow is drawn. The correctional equation shall be defined with the discrepancy between the results of measurements and calculations.

The transformation of the flexible curve caused by the H and F forces as well as M momentum can be calculated in the ζ-η system of coordinates with the application of the basic rules in stress analysis. The curve is regarded as a flat curve and a braced holder. Based on the theoretical considerations of the above-mentioned, the transformations, i.e. the change of the Ψ angle and the *u, v* movements shall be calculated as follows (based on Muttnyánszky 1981) in equation 16 a-c.

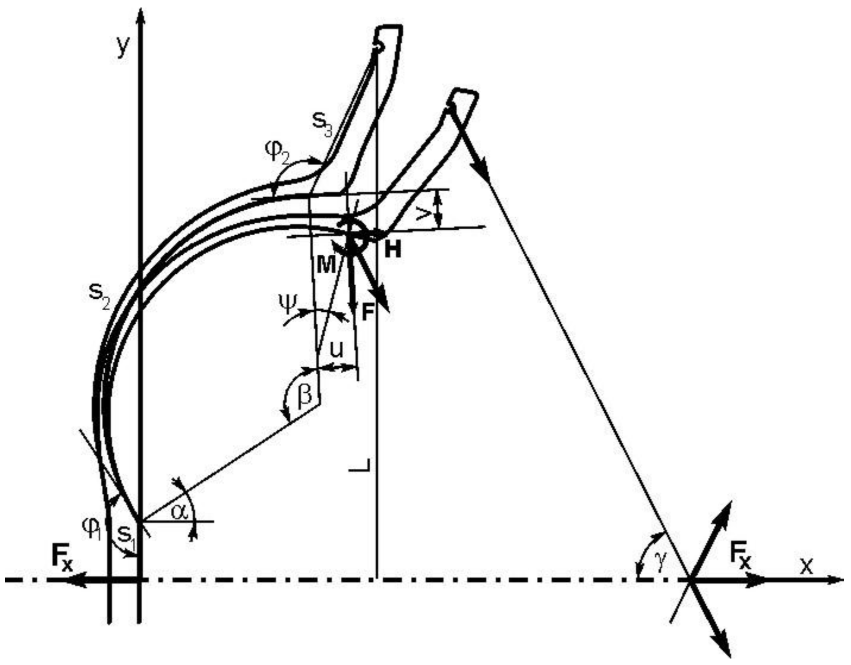


Figure 3
The theoretical constructed figure of a drawn bow

$$\psi = \frac{r}{IE}\left(a_1 M + a_2 rF + a_3 rH\right),$$
$$u = \frac{r^2}{IE}\left(b_1 M + b_2 rF + b_3 rH\right), \qquad (16\text{ a-c})$$
$$v = \frac{r^2}{IE}\left(c_1 M + c_2 rF + c_3 rH\right),$$

in which

$$a_1 = \beta,$$
$$a_2 = c_1 = \sin(\beta) - \beta\cos(\beta),$$
$$a_3 = b_1 = \beta\sin(\beta) + \cos(\beta) - 1,$$
$$b_2 = c_3 = 0{,}5 - \left[1{,}5\cos(\beta) + \beta\sin(\beta) - 1\right]\cos(\beta), \qquad (17\text{ a-f})$$
$$b_3 = \beta\left[0{,}5 + \sin^2(\beta)\right] + (1{,}5\cos(\beta) - 2)\sin(\beta),$$
$$c_2 = \beta\left[0{,}5 + \cos^2(\beta)\right] - 1{,}5\sin(\beta)\cos(\beta).$$

The transformation of the drawn bow is significant, therefore these transformations strongly affect the forces, the situation, direction and size of the momentum of the forces causing deformation. Because of this situation the geometry and the play of power forces of the bow shall be determined with iteration, i.e. the method of gradual approach.

The first step of iteration is to modify ζ_2 and η_2 as interpreted in the ζ-η. system of coordinates and defined as (P_2) that is the common point of the x_2 and y_2 coordinates of the flexible bow arm and the horn, with *u, v* and ψ transformation figures, which can most easily be read from a constructed figure. Further on, the new coordinates are defined with the knowledge of the calculated transformation during iteration:

$$\begin{aligned} \xi_2^* &= \xi_2 + u, \\ \eta_2^* &= \eta_2 - v. \end{aligned} \tag{18 a-b}$$

The familiar transformation of coordinates is used for the calculations between the ζ-η system of coordinates which are revolved with x-y and angle α of coordinates.

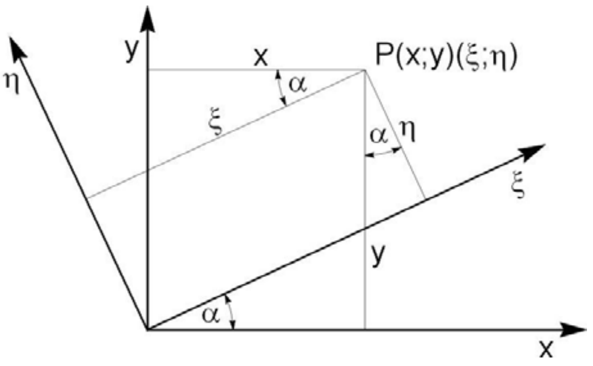


Figure 4

The equation for the calculation from the x-y system of coordinates to the ζ-η system:

$$\begin{bmatrix} \xi \\ \eta \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}. \tag{19}$$

And here is the reverse of the equation, i.e. the calculation is transferred from the ζ-η system of coordinates to the x-y system:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{bmatrix} \begin{bmatrix} \xi \\ \eta \end{bmatrix}. \tag{20}$$

With the knowledge of ζ_2^* and η_2^*, x_2^* and y_2^* as the new coordinates of P_2 shall be calculated with the application of the above-mentioned transformation of coordinates. After all P$_3$ as the new position of the common point of the bow horn and the string can be calculated as follows:

$$\begin{aligned} x_3^* &= x_2^* + s_3 \cdot \cos(3\pi/2 + \alpha - \beta - \varphi_2 - \psi) = x_2^* + s_3 \cdot \sin(\alpha - \beta - \varphi_2 - \psi), \\ y_3^* &= y_2^* + s_3 \cdot \sin(3\pi/2 + \alpha - \beta - \varphi_2 - \psi) = y_2^* - s_3 \cdot \cos(\alpha - \beta - \varphi_2 - \psi). \end{aligned} \qquad (21\text{ a-b})$$

Therefore the half angle of the string:

$$\gamma = \arcsin\left(\frac{y_3^*}{L}\right), \qquad (22)$$

and then the x_F coordinate of the introduction of force of the F_x pulling force:

$$x_F = x_3^* + \sqrt{L^2 - y_3^{*2}} \ . \qquad (23)$$

The calculation of the pulling force in the string:

$$F_1 = \frac{F_x}{2 \cdot \cos(\gamma)} . \qquad (24)$$

The same equation in a vector form:

$$\mathbf{F}_1 = \begin{bmatrix} F_1 \cos(\gamma) \\ -F_1 \sin(\gamma) \end{bmatrix} . \qquad (25)$$

The position vector between the P$_2$ and P$_3$ points:

$$\mathbf{r}_{23} = \begin{bmatrix} x_3^* - x_2^* \\ y_3^* - y_2^* \end{bmatrix} . \qquad (26)$$

The vector of M turning momentum from transferring force F$_1$ from P$_3$ point to point P$_2$:

$$\mathbf{M} = \mathbf{r}_{23} x \mathbf{F}_1 = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ r_x & r_y & 0 \\ F_1 \cos(\gamma) & -F_1 \sin(\gamma) & 0 \end{vmatrix} = -F_1 [r_x \sin(\gamma) + r_y \cos(\gamma)] \cdot \mathbf{k} \qquad (27)$$

from which the size of momentum shall be calculated as follows:

$$M = F_1[(x_3^* - x_2^*)\sin(\gamma) + (y_3^* - y_2^*)\cos(\gamma)]. \tag{28}$$

The forces loading the braced circularly curved holder:

$$H = F_1 \sin(\beta - \alpha - \gamma),$$
$$F = F_1 \cos(\beta - \alpha - \gamma). \tag{29 a-b}$$

After all the new figures of the ψ angle change and the u and v movements can be calculated with the (16 a-c) equations, then with the knowledge of this we can formulate the new coordinates of the common P_2 point of the flexible bow arm and the horn, while the iteration can be continued until the calculation has reached the appropriate margin of error.

4 The Energetical Analysis of the Measurements of the Bow

By means of the model, the parameters of the change of certain geometrical measurements on the energy accumulated in the bow, as the most typical characteristic of the application of the bow, can be analysed. When a bow is drawn, flexible energy accumulates in its structure, which gets mainly transferred to the arrow during shooting, while causing it to move. The characteristics of the bow are concluded from the relationship between the extent of the tension and the force that is necessary for it. See the characteristic equation below:

$$F = f(x), \tag{30}$$

in which Fx is the x direction force belonging to x distance (extension). The flexible energy accumulated in the bow can be calculated as follows:

$$E = \int_0^x f(x) \cdot \mathrm{d}x . \tag{31}$$

In the following part of this study the consequences of the individual alterations of each of the four geometrical characteristics of the Hungarian composite reflex bow shall be discussed. For the sake of better comparison, the maximum of the pulling force is defined as 200 N in every case, which as a matter of fact results in a deformation of different extent in the cases of bows of different sizes despite the fact that the cross section of the flexible bow arm (secondary momentum) and the material (flexibility modulus) have not changed. Regarding the characteristics of a real bow as standard, the measures are modified by -40, -20, +20 and +40%, then the characteristic curves are defined followed by the formulation of the energy of the bow.

The basic characteristics of the bow which are based on the measurements of the Hungarian bow known from archeological findings as well as the bow that was available for our research (these figures are later referred to as „standard" characteristics of the bow):

$2s_1$	=	112 mm
s_3	=	226 mm
$2L$	=	1260 mm
A	=	93 mm
B	=	367 mm
x_0	=	148 mm

First of all the characteristic curve of the bow is formulated with measuring and calculations:

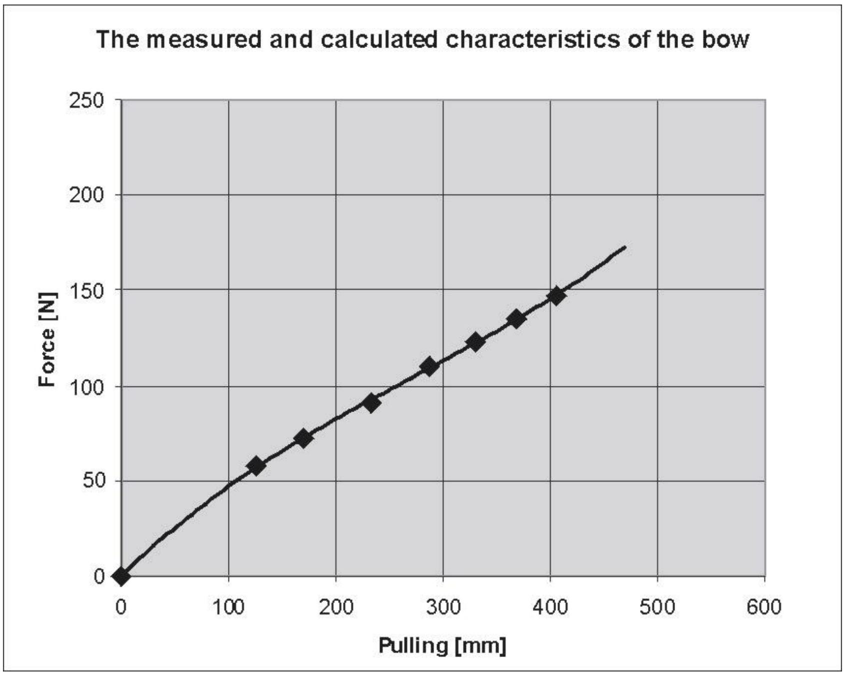


Figure 5
Characteristic curve of the bow

The applied equation for correction:

$$k = 100/(-0{,}000001 \cdot \Delta x^3 + 0{,}0008 \cdot \Delta x^2 + 0{,}097 \cdot \Delta x)\,, \tag{32}$$

in which *Δx* means the proportion of the draw in mm:

$$\Delta x = x_F - x_0 \,. \qquad (33)$$

When the *IE* product of multiplication is multiplied by the *k* correctional factor, the error concluded from the assumptions shall decrease.

The following characteristic curve shown in the next figure is the result if the s_1 size of the grip section is changed.

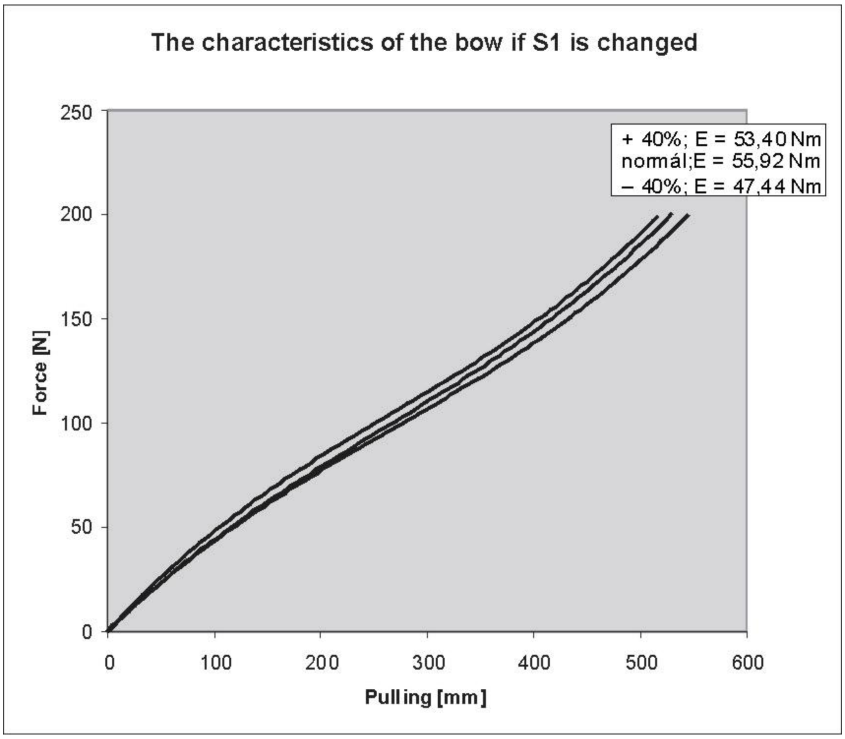


Figure 6

As it can clearly be seen in Figure 6, the alteration of the size of s_1 does not practically affect the static characteristics of the bow.

„A" measurement indicates the longest distance between the geometrical string of the curved bow arm and the bow arm itself. If „A" is changed, the following characteristic curves can be drawn as in Figure 7.

It can be seen well in Figure 7 that the characteristics of the bow hardly change if the curve is increased, however, if the curve is decreased, it results in a substantial modification of the characteristics. From the energetic point of view, the 20% decrease in the curve of the bow marked „normal" may result in some improvement.

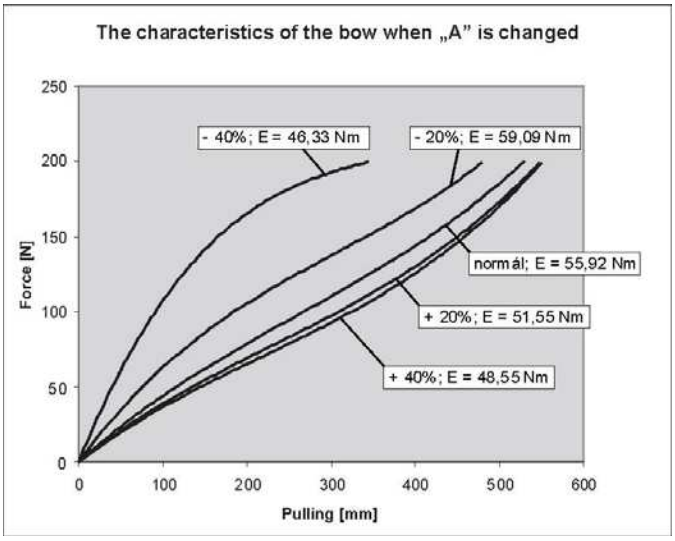


Figure 7

Changing the „B" measurements, i.e. the length of the longest string of the flexible bow arm may result in the following characteristic curves:

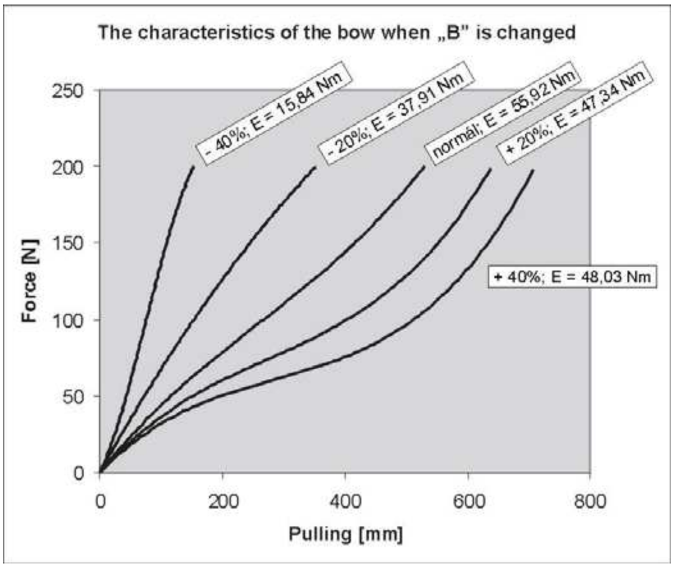


Figure 8

It can clearly be established if we look at the curves that the length of the flexible bow arm makes a substantial impact on the energy stored in the bow, therefore it influences the quality and the efficiency of the bow.

If the length of the so-called horn of the Hungarian bow (s_3 measurement) is modified, the characteristic curve will change according to Figure 9.

It is perhaps surprising what important role the horn plays in storing the energy in the bow. The horn is rigid and its deformation can be ignored, nevertheless it is a substantial characteristic element when it comes to the geometry and the functioning of the bow. The horn is responsible for the increase of the pulling length of the bow, which increases not only the velocity of the arrow and the energy that can be transmitted to the arrow, but it has also lead to a smaller size bow, which is one of the most outstanding features of reflex bows, as it is only the little size of the bow that makes horsing archery possible, therefore this characteristic had contributed greatly to the irresistable fighting manner of the conquering ancient Hungarians.

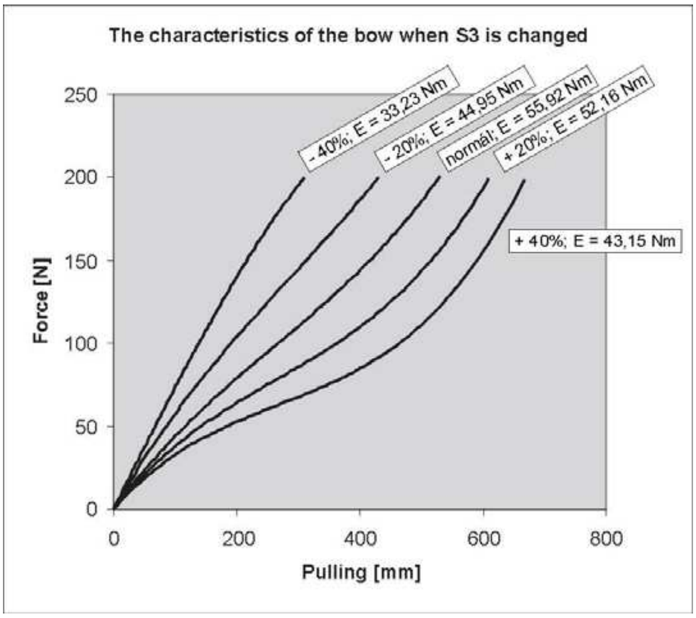


Figure 9
Change of characteristic curve

Finally Figure 10 summarizes the effect of the four altered geometrical characteristics of the Hungarian bow on the energy accumulated in the bow when it is affected by F=200 N force.

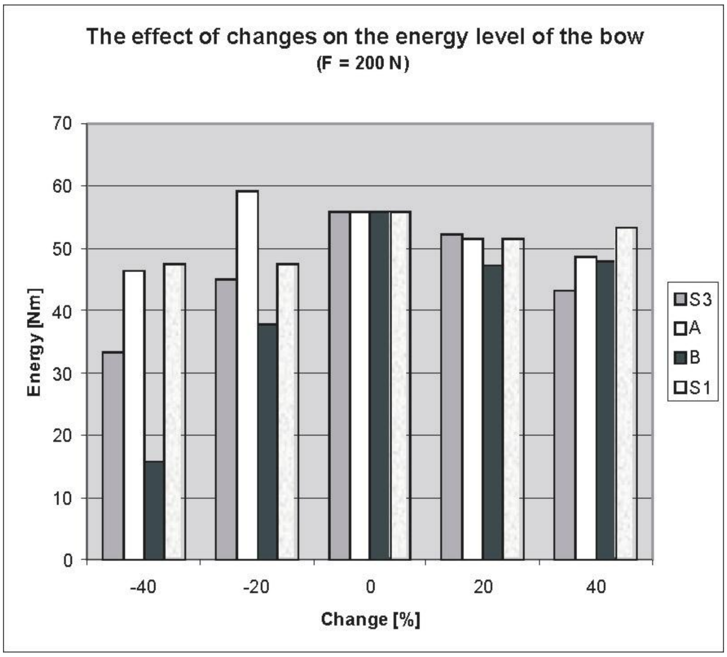


Figure 10

Conclusion

Based on the energetical analysis of the bow several conclusions can be drawn. It may be found surprising what important roles are played by the horns regarding the energy accumulated in the bow. Although the horn is rigid and its deformation can practically be ignored, it is still a relevant element in the geometry and the operation of the bow. Due to the horn the length of the extention of the bow substantially increases, which increases not only the acceleration path of the arrow and the transmittable energy but also results in the development of a small bow which means one of the most ingenious characteristics of reflexive bows since this small size makes their usage available for horse archery, moreover this characteristic contributed to the irresistable fighting manner of old Hungarians at the time of the Hungarian Conquest.

Finally the graph summarising the results of the various calculations shows that the analysed Hungarian bow has almost ideal measurements, which must have been the result of our ancestors' long experiments with the proportions of the bow throughout several centuries.

References

[1] Pauler Gyula – Szilágyi Sándor (szerk.): A magyar honfoglalás kútfői. A Magyar Tudományos Akadémia kiadása. Budapest, 1900

[2] Cs. Sebestyén Károly: Rejtélyes csontok népvándorláskori sírokban. Szeged, 1931

[3] Cs. Sebestyén Károly: A magyarok íja és nyila. Szeged, 1933

[4] Csallány Dezső: Kora-avarkori sírleletek. FA I-II. (1938–39), 121-180

[5] Győrffy György: A magyarok elődeiről és a honfoglalásról. Budapest, 1958

[6] Fábián Gyula: Archaeologia experimentalis. Honfoglaláskori magyar íj rekonstruálása. Természettudományi Közlöny 9, (1967) 98-101

[7] Fábián Gyula: The Hungarian composite. Journ. of the Soc. of Archer-Antiquaries 1970, 12-16

[8] U. Kőhalmi Katalin: A steppék nomádja lóháton, fegyverben. Budapest, 1972

[9] Fábián Gyula: Újabb adatok a honfoglaláskori íjászat kérdésköréhez. SZMMÉ 1980–81/1, 63-76

[10] Muttnyánszky Ádám: Szilárdságtan. Műszaki Könyvkiadó, Budapest, 1981.

[11] Fábián Gyula: A honfoglaláskori magyar íj és készítése. Nimród Fórum (A Nimród szakmai melléklete.) 1985. ápr. 1-11

[12] Fábián Gyula: Az avar domb kincse. Természet Világa 1985/5. sz. 211-214

[13] Szőllősy Gábor: Újabb adatok a népvándorláskori íjtípusok kérdésköréhez. JAMÉ 30-32 (1987–1989) 349-374

[14] Moravcsik Gyula: Az Árpád-kori magyar történet bizánci forrásai. Akadémiai Kiadó, Budapest, 1988

[15] László Gyula: Árpád népe. Helikon Kiadó, Budapest, 1988

[16] Szőllősy Gábor: Különböző íjtípusok mechanikai jellemzőinek kísérleti vizsgálata. Doktori értekezés. Kézirat. Budapest, 1995

[17] Fodor István – Diószegi György – Legeza László: Őseink nyomában. Magyar Könyvklub – Helikon Kiadó, Budapest, 1996

Robust Crane Control

Ing. Marek Hičár, Ing. Juraj Ritók, PhD

Department of Electrical Drives And Mechatronics, Department of Design, Transport and Logistics, Technical University in Košice, Letna 9/B, 042 00 Kosice, Slovak Republic, Tel.: +421-55-6022268, 6022503
E-mail: hicarm@hron.fei.tuke.sk, juraj.ritok@tuke.sk

Abstract: The paper presents robust crane design by asynchronnous motor with frequency convertor at ensuring system robustness against load weight and rope length variation. Exact position control and elimination of swinging in the final position are required too. Firstly were assemblied mathematical models of main crane components: crab, bridge and uplift by real model of double beamed bridge experimental crane. Was designed robust control for defined interval variation of weight and rope length for real crane. Load weight and swinging are determined by estimators. Finally measured results gained on a laboratory crane are introduced.

Keywords: robust control, crane crab and bridge, uplift, poles region assignment method, observer, estimator, swinging model

1 Introduction

Control of main system drives control has to ensure the most effective and exact motion for crane crab, bridge motion and uplift. This control of crane crab and bridge includes two the most important conditions of exact motion trajectory and forbidden swinging in the final position. Crane systems using today keep precisious positioning but not elimination of the load swinging. Conditions have to be realized for different load weights and lengths of hanging rope. System robustness against change of load weight was designed by Ackermann's by finding suitable feedbacks from robust areas which provides desired properties of the whole system. Rope length belongs to variable parameters what undertakes checking of control design for stability. Switching robust structure feedbacks (areas) for covering total tonnage and all rope lenghts was done by robust subareas which provide robustness against load weight and rope length variaton. Load swinging observer in crane crab and bridge direction were designed for reason of elimination electromechanical load swinging measurement for zero deviation control. Next, load weight observer identified real load weight on the crane hook which can vary between minimum (hook with tackle) and maximum crane tonnage. All positive solutions of crane drives design are applied for real experimental bridge crane located at experimental laboratory Department of Construction, Transport and logistics at Technical University in Kosice. Load weight estimator was established for identification of real load for robust control. Our tendency was approaching results from swinging model with measurement parameters what demostrates application of subsystem models in connection with real objects.

2 Poles Region Assignment Method

Poles region assignment method has ambition to get values of robust controllers (finding of load weight location in relation to rope lengths) where load swinging will be damped. Control design is procedure for finding feedback vector so that poles of characteristical polynom should be located in Γ - plane at parameters variation (load weight and rope length) (Figure 1). After matrix multiplying in formula (1) and comparison with right side will be expressed pair of robust controllers r_1, r_2, while α is generalized frequency. Graphical draw of $r_1 = f(r_2)$ at load weight changing helps to choose correct controllers parameters from areas of figures 3 and 5 [1, 2].

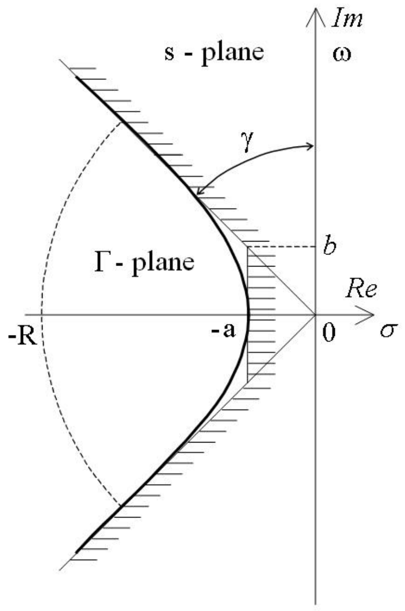


Figure 1
Location of Γ - plane

Ackermann's condition:

$$\begin{bmatrix} 1 & 2\alpha & 3\alpha^2 - \frac{b^2}{a^2}\alpha^2 + b^2\alpha^3 & -\frac{4b^2}{a^2}\alpha^3 + 4b^2\alpha & \dots & d_{n-1} \\ 0 & 1 & 2\alpha & 3\alpha^2 - \frac{b^2}{a^2}\alpha^2 + b^2\alpha^3 & \dots & d_{n-2} \end{bmatrix} \mathbf{a}(\mathbf{r}_1, \mathbf{r}_2) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \qquad (1)$$

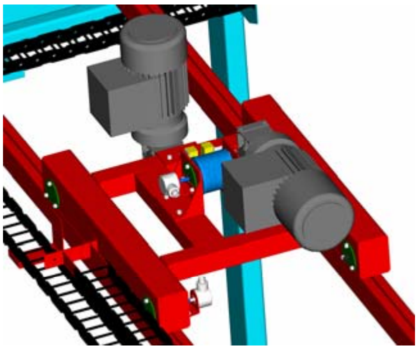


Figure 2

Crane crab

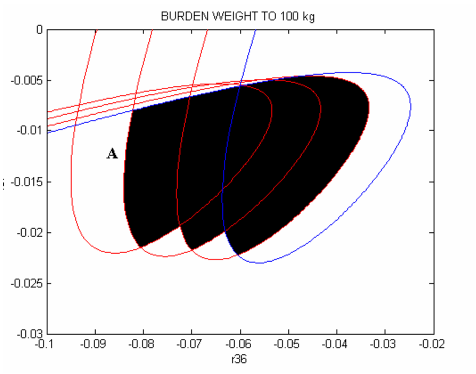


Figure 3

Robust controllers design for crab

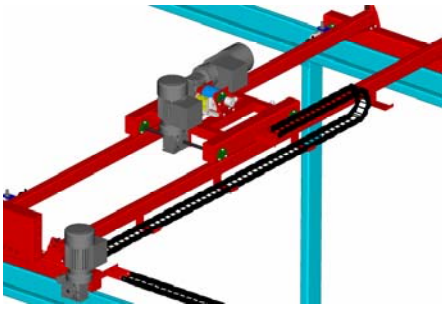


Figure 4

Crane bridge

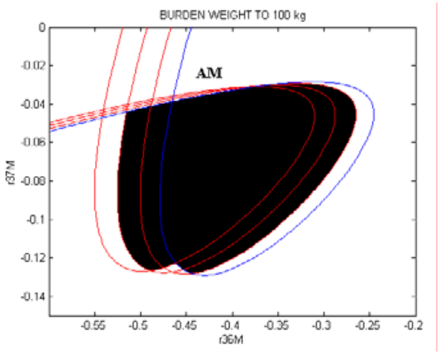


Figure 5

Robust controllers design for bridge

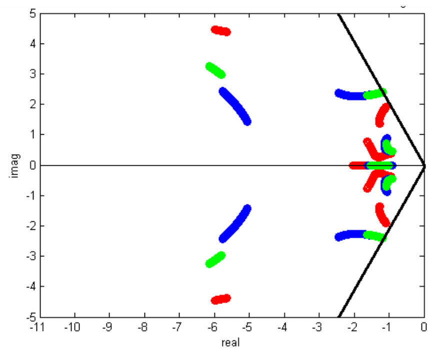


Figure 6

Poles motion at weight change

Robust crab controllers (r_{36} , r_{37}) and bridge (index M) ensure load swinging control and its speed at all crane tonnage. Figure 6 represents keeping poles trajectory of characteristical polynom in allowed area of stability and damping at load weight variation (0,100) kg. It is same for poles trajectory at rope length change (0,5;2,5) m [3].

3 Crane Crab and Bridge Model

Crane crab and bridge serve for load transport (separately or all at once) with weight m_G from initial position x_{K1} to the final position x_{K2} [3].

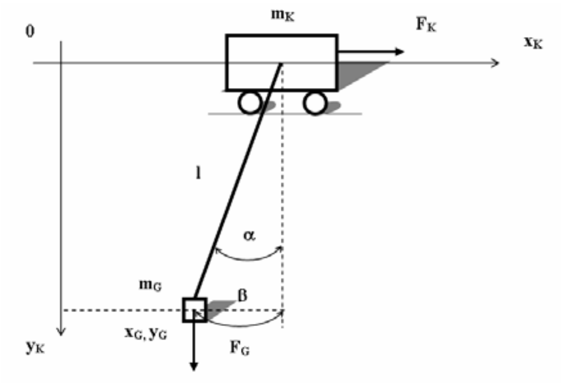


Figure 7
Crane rab and bridge model

Differential and algebra formulas for description of mechanical crane crab and bridge part: $m_K \ddot{x}_K = F_K + F_G \sin\alpha$, $m_G \ddot{x}_G = -F_G \sin\alpha$, (2)

$$m_G \ddot{y}_G = m_G g - F_G \cos\alpha, \tag{3}$$

$$x_G = x_K + l\sin\alpha,\ y_G = l\cos\alpha. \tag{4}$$

$$\ddot{x}_K = \frac{3p}{2}\frac{L_h}{1+\sigma_2}\frac{j}{m_K r}\frac{1}{\left(\frac{J_m}{p}\frac{j^2}{r^2}\frac{1}{m_K}+1\right)} i_{2m} i_{1y} + \frac{m_G}{m_K}\frac{g}{l}\frac{1}{\left(\frac{J_m}{p}\frac{j^2}{r^2}\frac{1}{m_K}+1\right)}\beta;$$

$$\ddot{\beta} = -\frac{3p}{2}\frac{L_h}{1+\sigma_2}\frac{j}{m_K r} i_{2m} i_{1y} + \frac{J_m}{p}\frac{j^2}{r^2 m_K}\ddot{x}_K - \frac{\left(1+\frac{m_G}{m_K}\right)}{\frac{l}{g}}\beta. \tag{5, 6}$$

Formulas 5 and 6 describe crab (bridge) acceleration $\ddot{x}_K$ and swinging acceleration $\ddot{\beta}$ in its direction where r is transmission radius. α is angle of swinging and β is transmitted angle to the meters.

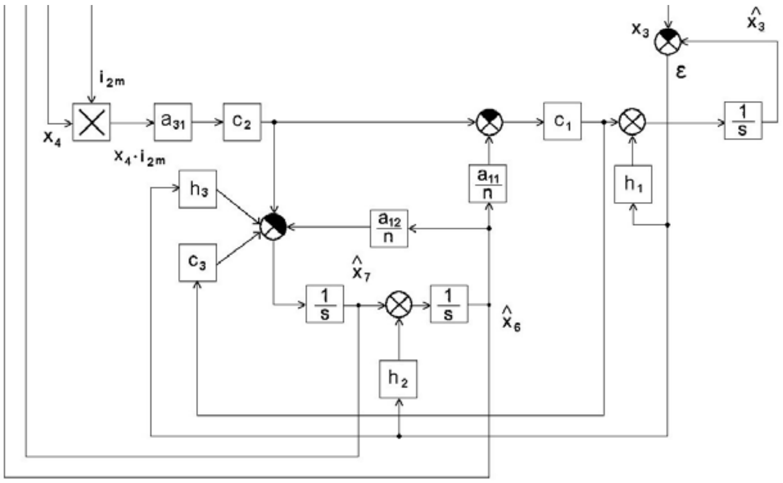


Figure 8

Load swinging observer with output $\hat{x}_6$ and its speed $\hat{x}_7$

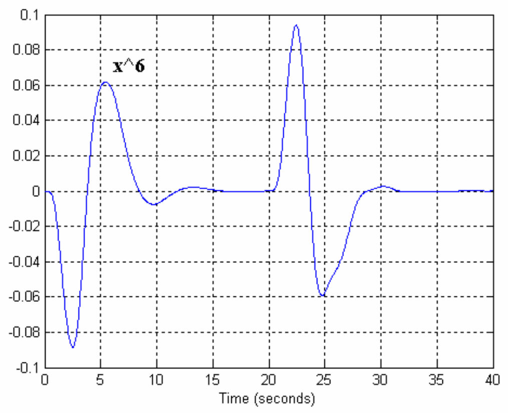


Figure 9
Load swinging in crab direction

Simulations in MATLAB Simulink on figure 9 and 10 are time respond of observer load swinging in crane crab and bridge direction. There was accepted maximum overswing in the final position 0.5 cm from practical reason by low toughness of hanging rope.

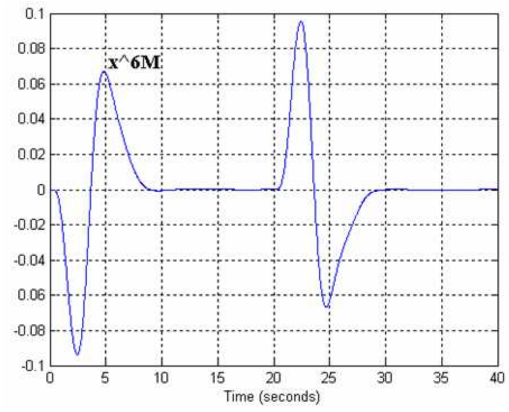


Figure 10
Load swinging in bridge direction

Load swinging control in simulation model is ensured by feedback from load swinging $\hat{x}_6$ ($\hat{x}_{6M}$) [m] and its speed $\hat{x}_7$ ($\hat{x}_{7M}$) [ms^{-1}]. On Figure 10 is observed zero swinging at the end of transport.

4 Crane Uplift Model

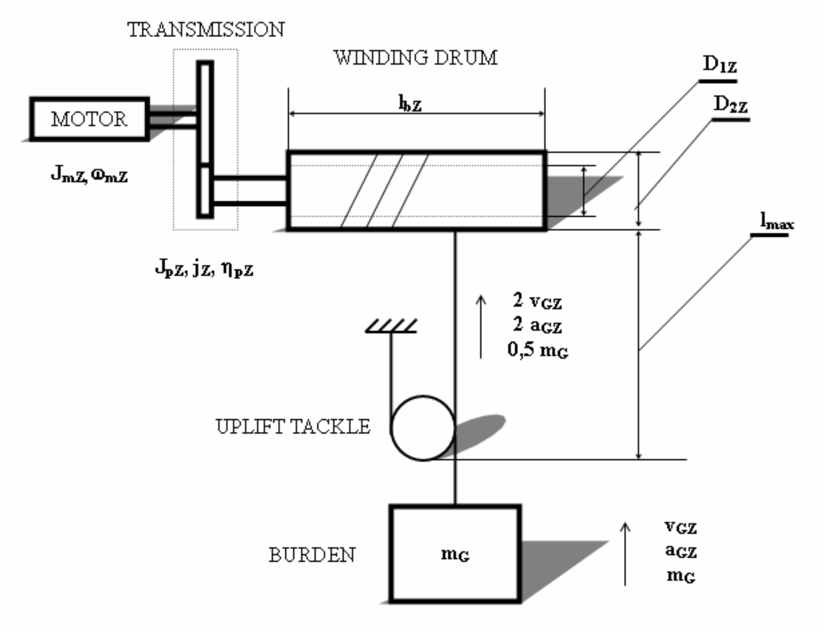


Figure 11
Crane uplift model

Torque formula and total moment of inertia for crane uplift:

$$M_{mZ} - M_{GZ} = J_{CZ} \frac{d\omega_{mZ}}{dt}, \tag{7}$$

where M_{mZ} - motor moment, M_{GZ} - load moment, J_{CZ} - total moment inertia, J_{mZ} - motor moment inertia, J_{CbZ} - drum moment inertia, J_{CGZ} - load moment inertia.

Simulation of weight observer output provides real weight on the hook and this information is needed for robust controllers switching by defined subarea.

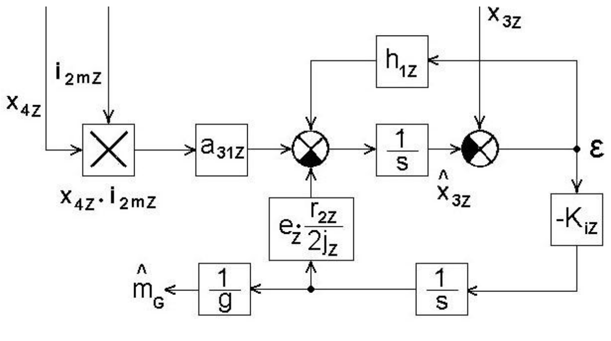


Figure 12
Weight observer with load weight output

On the Figure 13 is shown simulation crane bridge motion x_{5M} [m] after sellecting robust controllers according to identified load weight $\hat{m}_G$ [kg]. Real rope length is gained from uplift model for next setting robust controllers.

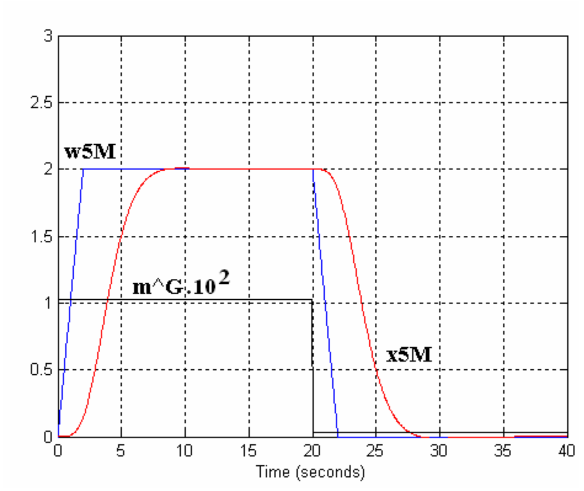


Figure 13
Crane bridge trajectory and identified weight

5 Measured Processes at Experimental Bridge Crane

Bridge crane was experimentaly indentified by ARX model and was acquired linearized transfer functions of crab, bridge and uplift. Swinging in crab and bridge direction was identified by OE model. Robustness of real experimental crane model was ensured by switching structure of robust controllers for covering load weight and rope length variations.

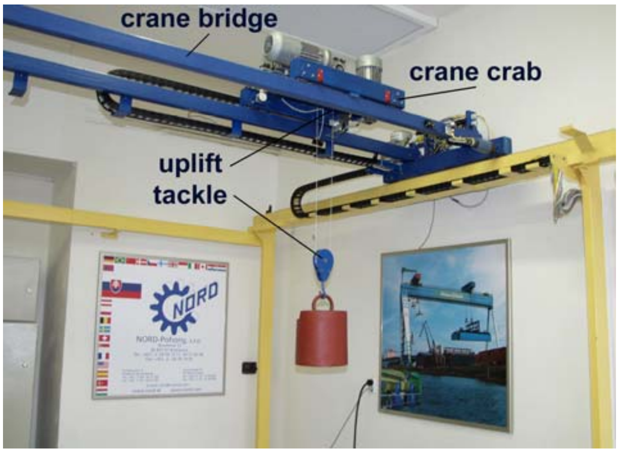


Figure 14
Experimental bridge crane

Swinging sensor design consists of two each other perpendiculary rotary rheostats which measure deviation in crane crab and bridge direction.

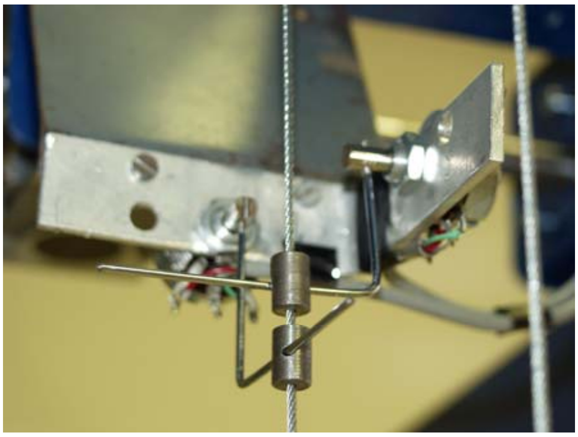


Figure 15
Swinging sensor

On the Figure 16 is shown identification of load weight m_{rG} [kg] by estimator based on load uplift current. Figure 17 represents matching of real load swinging in crane crab direction x_{6r} [m] with swinging model x_{6m} [m] which realized actuating signal to the control. Rope length is l = 1.1 m. Measured time responses on the Figure 18 is time response of contemporary crab x_{5r} and bridge x_{5rM} motion at rope length l = 2.3 m. Load swinging is plotted on the Figure 19 at condition of zero swinging in the final position.

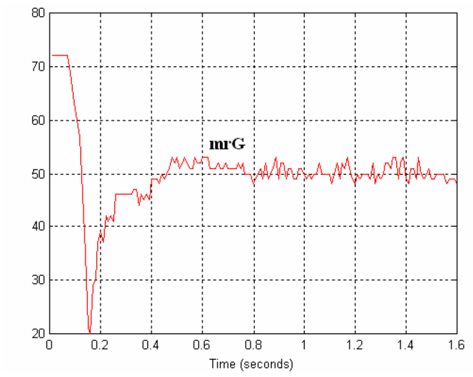


Figure 16

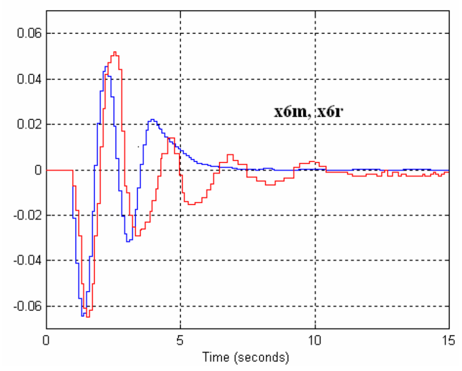


Figure 17

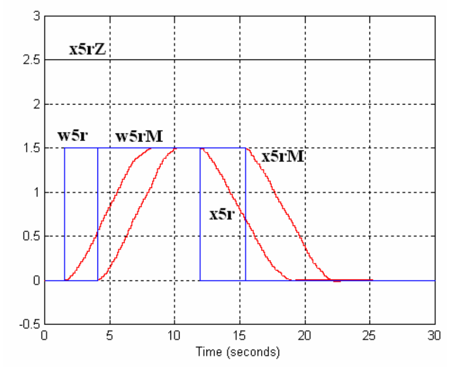


Figure 18

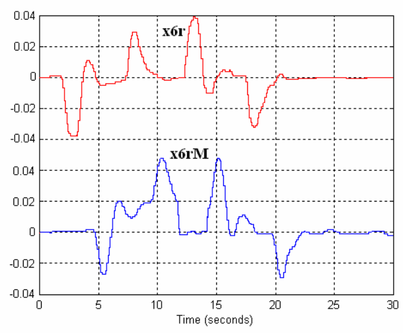


Figure 19

Project Experimental bridge crane was solved in cooperation with Department of Construction, Transport and Logistics and Department of Cybernetics and Artificial Intelligence.

Conclusions

Robust crane control as system with variable parameters was designed by Ackermann method of poles region assignment where by defined algorithm were obtain robust controllers for ensuring stability and damping at load weight and rope length variation in user range. Correct robust design was confirmed by measurement with included weight and model load swinging estimator at disallowed load swinging in the final reference position in crane crab and bridge direction.

Referencies

[1] Ackermann, J.: Parameter Space Design of Robust Control Systems, IEEE Trans. On Automatic Control, 1980

[2] Leonhard, W.: Control Of Electrical Drives, Springer Vlg., Berlin, 1985

[3] Hičár, M. : Written work to the academic dissertation exam, Robust crane crab control, Košice 2001 (in Slovak)

[4] Ritók, J., Bigoš, P.: Automate crane in logistics system, In: International conference Logistics & Transport, Vysoké Tatry, 2001 (in Slovak)

[5] Y. Sakawa and H. Sano. Nonlinear model and linear robust control of overhead travelling cranes. Nonlinear Analysis, 30(4):2197{2207, 1997

Level Crossing Probabilities of the Ornstein – Uhlenbeck Process

Dr. József Dénes

Institute of Informatics and Mathematics, Faculty of Light Industry, Budapes Tech
Department of Science, University of West Hungary
denes.jozsef@nik.bmf.hu

Abstract: The Ornstein Uhlenbeck process is a Gaussian process X_t *with independent increments and autocorrelation* $E(X_t X_{t+s}) = \frac{e^{-|s|}}{2}$*. First the Laplace transform of the probability density* $P(X_t = x | X_0 = p)$ *is computed. Using this, the Laplace transform of* X_t *first time reaching a given value* x *is derived. It is proved that these results agree with the special case derived earlier by Bellman and Harris (Pacific J. Math. 1, 1951).*

1 Definitons

The Ornstein Uhlenbeck process is a stationary Gaussian-Markov process X_t such that the joint distribution of $X_{t1,} X_{t2} \dots X_{tm}$ is a gaussian and is dependent only on the differences $t_j - t_i$ where $i < j$ and the autocorelation function is given by

$$E(X_s \cdot X_{s+t}) = \frac{1}{2} e^{-|t|} \tag{1.1}$$

$$EX_t = 0 \text{ and } EX_t^2 = \frac{1}{2}. \tag{1.2}$$

Let X be a random vector with normal distribution, then the density of its probability distribution is:

$$\frac{1}{2\pi|\Sigma|} e^{-\frac{1}{2} X^T \Sigma^{-1} X}$$

where $X = \begin{pmatrix} X \\ Y \end{pmatrix}$ and Σ is the correlation matrix:

$$\begin{pmatrix} \rho_1 & \sigma \\ \sigma & \rho_2 \end{pmatrix}$$

with $\rho_1 = EX^2, \rho_2 = EY^2, \sigma_1 = EXY$ and $|\Sigma| = \rho_1\rho_2 - \sigma^2$. Clearly

$$\Sigma^{-1} = \frac{\begin{pmatrix} \rho_2 & -\sigma \\ -\sigma & \rho_1 \end{pmatrix}}{\rho_1\rho_2 - \sigma^2}.$$

Hence the joint probability density

$$P(X = x, Y = y) = \frac{1}{2\pi\sqrt{\rho_1\rho_2 - \sigma^2}} \exp\left(- \frac{\rho_2 x^2 - 2\sigma xy + \rho_1 y^2}{2(\rho_1\rho_2 - \sigma^2)} \right).$$

It follows from here that

$$P(Y = y | X = x) = \frac{\frac{1}{2\pi\sqrt{\rho_1\rho_2 - \sigma^2}} \exp\left(- \frac{\rho_2 x^2 - 2\sigma xy + \rho_1 y^2}{2(\rho_1\rho_2 - \sigma^2)} \right)}{\frac{e^{-\frac{x^2}{2\rho_1}}}{\sqrt{2\pi\rho_1}}}$$

$$= \frac{1}{\sqrt{2\pi\frac{\rho_1\rho_2 - \sigma^2}{\rho_1}}} \exp\left(- \frac{\left(y - \frac{\sigma^2}{\rho_1} x \right)^2}{2\frac{\rho_1\rho_2 - \sigma^2}{\rho_1}} \right).$$

Applying this to what concerns us, the Ornstein-Uhlembeck process, we can determine the probability density $P(X_t = x \mid X_0 = p)$.

Clearly

$$\rho_1 = \rho_2 = \frac{1}{2}, \sigma = \frac{e^{-t}}{2} \text{ so } \frac{2(\rho_1\rho_2 - \sigma^2)}{\rho_1} = \frac{2\left(\frac{1}{2}\frac{1}{2} - \frac{e^{-2t}}{4}\right)}{\frac{1}{2}} = 1 - e^{-2t}. \ \frac{\sigma}{\rho_1} = e^{-t}$$

Hence:

$$P(X_t = x \mid X_0 = p) = \frac{e^{-\frac{\left(x - pe^{-t}\right)^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi\left(1 - e^{-2t}\right)}}. \tag{1.3}$$

We shall denote this with $P(t, p, x)$ or $P(p, x)$ and call it the fundamental function. The special cases $p = 0$ and $x = 0$ are important also:

$$P(X_t = x \mid X_0 = 0) = \frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi\left(1 - e^{-2t}\right)}}. \tag{1.4}$$

$$P(X_t = 0 \mid X_0 = p) = \frac{e^{-\frac{p^2e^{-2t}}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi\left(1 - e^{-2t}\right)}}. \tag{1.5}$$

By simple substitution it is easy to prove that (1.3) satisfies the forward equation:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\frac{\partial^2 u}{\partial x^2} + x\frac{\partial u}{\partial t} + u$$

and the backward equation:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\frac{\partial^2 u}{\partial p^2} - p\frac{\partial u}{\partial p}.$$

This also implies that (1.4) satisfies the forward equation and (1.5) satisfies the backward equation.

2 The Laplace Transforms of the Fundamental Functions

Since both $\frac{e^{-\frac{\left(x-pe^{-t}\right)^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi\left(1-e^{-2t}\right)}}$ and $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi\left(1-e^{-2t}\right)}}$ satisfy the $u_t = \frac{1}{2}u_{xx} + u + xu_x$

forward equation their Laplace transforms must satisfy the $sU = \frac{U_{xx}}{2} + U + xU_x$ second order ordinary differential equation, that is the equation

$$U'' + 2xU' + 2(1-s)U = 0 \tag{2.1}$$

To find the solutions of (2.1) let us consider the confluent hypergeometric equation

$$xy'' = +(c-x)y'0 - ay = 0 \tag{2.2}$$

The two solutions of this are the:

$$_1F_1(a,c;x) = 1 + \frac{a}{c}\frac{x}{1!} + \frac{a(a+1)x^2}{c(c+1)2!}\dots$$

and $x^{1-c}\,{}_1F_1(a+1-c, 2-c; x)$ Kummer functions. Let us consider the following transformation of (2.2) $u = y(kx^2)$ where k is an arbitrary nonzero constant.

Clearly:

$$u = y(kx^2)$$
$$u' = 2kxy'$$
$$u'' = 2ky' + 4k^2x^2y''.$$

Hence:

$$y = u$$
$$y' = \frac{u'}{2kx}$$
$$y'' = \frac{u'' - \frac{u'}{x}}{4k^2x^2}$$

Substituting these into (2.2) gives:

$$\frac{kx^2\left(u'' - \frac{u'}{x}\right)}{4k^2x^2} + (c - kx^2)\frac{u'}{2kx} - au = 0$$

which in turn, after some simplification becomes:

$$u'' + \left(\frac{2c-1}{x} - 2kx\right)u' - 4kau = 0.$$

Putting $c\frac{1}{2}$ gives: $u'' - 2kxu' - 4kau = 0.$

Let us compare this with (2.1)

$$U'' + 2xU' + 2(1-s)U = 0$$
$$-2k = 2$$
$$-4ka = 2(1-s).$$

Hence we get for k and for $ak = -1$ and $a = \frac{1-s}{2}$. Therefore the solutions of (2.1) are $F_1 = F\left(\frac{1-s}{2}, \frac{1}{2}; -x^2\right)$ and $F_2 = xF\left(1 - \frac{s}{2}, \frac{3}{2}; -- x^2\right)$.

Now we are in the position to determine the Laplace transform of $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$.

Clearly it must be of the form $AF_1 + xBF_2$ where A and B some constans. To this end Laplace transform will be evaluated for some special cases. The Laplace transform of $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ is clearly:

$$\int_0^{\infty} \frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)} e^{-st} dt.$$

Writing t instead of e^{-t} transforms it into a Mellin type integral:

$$\int_0^{1} \frac{e^{-\frac{x^2}{1-t^2}}}{\sqrt{\pi}\left(1-t^2\right)} t^{s-1} dt.$$

Substituing $\sqrt{t}$ instead of t yields

$$\frac{1}{2\sqrt{\pi}} \int_0^{1} \frac{e^{-\frac{x^2}{1-t}}}{\sqrt{1-t}} t^{\frac{s}{2}-1} dt.$$

For $x = 0$ this becomes the beta funciton type integral:

$$\frac{1}{2\sqrt{\pi}} \int_0^{1} \frac{t^{\frac{s}{2}-1}}{\sqrt{1-t}} dt = \frac{1}{2\sqrt{\pi}} B\left(\frac{1}{2}, \frac{s}{2}\right) = \frac{\Gamma\left(\frac{1}{2}\right)\Gamma\left(\frac{s}{2}\right)}{2\sqrt{\pi}\Gamma\left(\frac{1+s}{2}\right)}.$$

Hence

$$A = \frac{\Gamma\left(\frac{s}{2}\right)}{2\Gamma\left(\frac{s+1}{2}\right)}.$$

Clearly A is the Laplace transform of $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$. To determine the value of B let us consider the x derivative of the Laplace transform, which is:

$$-\int_0^{1} \frac{x e^{-\frac{x^2}{1-t}}}{\sqrt{\pi}(1-t)^{\frac{3}{2}}} t^{\frac{s}{2}-1} dt.$$

In the present case we cannot take the $x \to 0$ limit by simply substituing $x \to 0$ for x because $\dfrac{xe^{-\frac{x^2}{t}}}{\sqrt{\pi} - t^{\frac{3}{2}}}$ does not converge uniformly to 0 as $x \to 0$ .In fact it is a "delta function type function", its integral being

$$\int_0^1 \frac{xe^{-\frac{x^2}{t}}}{\sqrt{\pi}t^{\frac{3}{2}}} dt = 1.$$

For it is know from theory of the heat equation that, for an arbitrary continous function $f(t)$

$$\lim_{x\to 0} \int_0^t \frac{x}{\sqrt{\pi}} \frac{e^{-\frac{x^2}{t-r}}}{(t-r)^{\frac{3}{2}}} f(r)dr = \lim_{x\to 0} \int_0^t \frac{x}{\sqrt{\pi}} \frac{e^{-\frac{x^2}{r}}}{r^{\frac{3}{2}}} f(t-r)dr = f(t).$$

Hence in the present case:

$$-\lim_{x\to 0} \int_0^1 \frac{xe^{-\frac{x^2}{1-t}}}{\sqrt{\pi}(1-t)^{\frac{3}{2}}} t^{\frac{s}{2}-1} dt = t^{\frac{s}{2}-1}\Big|_{t=1} = -1,$$

thus $B = -1$. Therefore the The Laplace transform of $\dfrac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ is

$$AF_1 - xF_2 = \frac{\Gamma\left(\frac{s}{2}\right)}{2\Gamma\left(\frac{s+1}{2}\right)} F\left(\frac{1-s}{2}, \frac{1}{2}; -x^2\right) - xF\left(1 - \frac{s}{2}, \frac{3}{2}; -x^2\right).$$

Now we compute the The Laplace transform of $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$. It has been shown that it statisfies the backward equation $u_t = -pu_p + \frac{u_{pp}}{2}$. Therefore its Laplace transform is the solution of the second order linear differential equation $sU = -pU_p + \frac{U_{pp}}{2}$ that is of the equation

$$U'' + 2pU' + 2sU = 0$$

Now the solution of $u'' - 2kxu' - 4kau = 0$ are $F\left(a, \frac{1}{2}; kx^2\right)$ and $xF\left(a + \frac{1}{2}, \frac{3}{2}; kx^2\right)$.

Comparing the two equations we get for k

$$2k = 2$$
$$4ka = 2s$$

that is $k = 1$ and $a = \frac{s}{2}$. Thus the Laplace transform must be the linear combination of $G_1 = F\left(\frac{s}{2}, \frac{1}{2}; p^2\right)$ and $pG_2 = pF\left(\frac{1+s}{2}, \frac{3}{2}; p^2\right)$. To find the conficciens of G_1 and pG_2 let us inspect the Laplace transform itself. $\int_0^\infty \frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)} e^{-st} dt.$

Writing t instead of e^{-t} it transforms again into the Mellin type integral:

$$\int_0^1 \frac{e^{-\frac{p^2 t^2}{\left(1-t^2\right)}}}{\sqrt{\pi}\left(1-t^2\right)} t^{s-1} dt.$$

Substituing $\sqrt{t}$ instead of t yields

$$\frac{1}{2\sqrt{\pi}}\int_0^1 \frac{e^{-\frac{p^2t^2}{1-t}}}{\sqrt{1-t}}\, t^{\frac{s}{2}-1}\,dt.$$

Again putting $p = 0$ this becomes:

$$\frac{1}{2\sqrt{\pi}}\int_0^1 \frac{t^{\frac{s}{2}}}{\sqrt{1-t}}\,dt = A.$$

The coefficient of pG_2 can be evaluated the same way as was done for $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ and it is found to be again -1. Thus the Laplace transform of $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ is

$$AG_1 - pG_2 = AF\left(\frac{s}{2}, \frac{1}{2}; p^2\right) - pF\left(1 - \frac{1+s}{2}, \frac{3}{2}; p^2\right).$$

The above result can be arrived at directly from the Laplace transform of $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$.

To this end let us inspect

$$\int_0^\infty \frac{e^{-\frac{p^2e^{-t2}}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}e^{-st}dt$$

using

$$\frac{p^2e^{-2t}}{1-e^{-2t}} = \frac{p^2}{1-e^{-2t}} - p^2.$$

This becomes $e^{p^2}\int_0^\infty \frac{e^{-\frac{p^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}e^{-st}dt$ and the integra here is of the same form as of the Laplace transform of $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ except we have p instead of x.

Therfore the Laplace transform of $\frac{e^{-\frac{x^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ is

$$e^{p^2}\left(\frac{\Gamma\left(\frac{s}{2}\right)}{2\Gamma\left(\frac{s+1}{2}\right)}F\left(\frac{1-s}{2},\frac{1}{2};-p^2\right)-pF\left(1-\frac{s}{2},\frac{3}{2};-p^2\right)\right)$$

Applying Kummer's formula $F(a,c;x)=e^xF(c-a,c;x)$ we get for the Laplace transform of $\frac{e^{-\frac{p^2e^{-2t}}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$

$$\frac{\Gamma\left(\frac{s}{2}\right)}{2\Gamma\left(\frac{s+1}{2}\right)}F\left(\frac{s}{2},\frac{1}{2};p^2\right)-pF\left(1-\frac{1+s}{2},\frac{3}{2};p^2\right).$$

3 The Laplace Transforms of $\frac{e^{-\frac{(x-pe^{-t})^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$

We have seen that the $\frac{e^{-\frac{(x-pe^{-t})^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ fundamental function satisfies both the forward and backward equations, therefore its Laplace transform must satisfy both of the ordinary differential equations:

$$U'' + 2pU' + 2(1-s)U = 0 \tag{3.1}$$

$$U'' - 2pU' - 2sU = 0. \tag{3.2}$$

Because of (3.1) must be of the form: $HF_1 + KxF_2$, where H and K must be some linear combinations of G_1 and pG_2 since it satisfies (3.2) as well. Let us observe that $\frac{e^{-\frac{(x-pe^{-t})^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ is analytic in x for all values p and t except when $t = 0$ and $x = p$, in the latter case it is undefined. Therefore its Laplace transform is analytic in the $x \le p$ domain as well. Putting $x = 0$ in $\frac{e^{-\frac{(x-pe^{-t})^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ gives $\frac{e^{-\frac{p^2e^{-t2}}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)}$ and we have seen that its Laplace transform is $AG_1 - pG_2$, so $H = AG_1 - pG_2$ (when $x \le p$). The determination of K is more involved. Differentiating the fundamental function by x gives:

$$\frac{2pe^{-t}e^{-\frac{p^2e^{-t2}}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)^{\frac{3}{2}}} = e^{p^2}\,\frac{2pe^{-t}e^{-\frac{p^2e^{-t2}}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)^{\frac{3}{2}}}. \tag{3.3}$$

Clearly the coefficient K is the Laplace transform of (3.3). To evaluate it let us compute the following convolution integral:

$$e^{p^2}\frac{2pe^{-t}e^{-\frac{p^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi}\left(1-e^{-2t}\right)^{\frac{3}{2}}}*\frac{1}{\sqrt{\pi\left(1-e^{-2t}\right)}}. \tag{3.4}$$

It has been shown that the Laplace transform of the second factor in (3.4) is A, so the Laplace transform of (3.3) is the Laplace transform of (3.4) divided into A. Next we evaluate (3.4):

$$\int_0^t e^{p^2}\frac{2pe^{-r}e^{-\frac{p^2}{\left(1-e^{-2r}\right)}}}{\sqrt{\pi}\left(1-e^{-2r}\right)^{\frac{3}{2}}}\frac{1}{\sqrt{\pi\left(1-e^{-2(t-r)}\right)}}dr =$$

putting r for e^{-r} yields:

$$\int_T^1 e^{p^2}\frac{2pre^{-\frac{p^2}{\left(1-r^2\right)}}}{\sqrt{\pi}\left(1-r^2\right)^{\frac{3}{2}}}\frac{1}{\sqrt{\pi\left(r^2-T^2\right)}}dr =$$

where $T=e^{-r}$. Substituting $\sqrt{r}$ for r gives:

$$e^{p^2}\int_{T^2}^1\frac{pe^{-\frac{p^2}{(1-r)}}}{\sqrt{\pi}(1-r)^{\frac{3}{2}}}\frac{1}{\sqrt{\pi\left(r^2-T^2\right)}}dr$$

$$=e^{p^2}\int_0^{1-T^2}\frac{pe^{-\frac{p^2}{\left(1-T^2-r\right)}}}{\sqrt{\pi}\left(1-T^2-r\right)^{\frac{3}{2}}}\frac{1}{\sqrt{\pi r}}dr$$

$$=e^{-p^2}\cdot\frac{pe^{-\frac{p^2}{t}}}{\sqrt{\pi t^3}}*\frac{1}{\sqrt{\pi t}}\bigg|_{t=1-T^2}=\frac{e^{-\frac{p^2e-2t}{(1-2^{-2t})}}}{\sqrt{\pi(1-2^{-2t})}}.$$

Thus we have for the coefficient $K=\dfrac{AG_1-pG_2}{A}$. Hence the Laplace transform of $\dfrac{e^{-\frac{(x-pe^{-t})^2}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi\left(1-e^{-2t}\right)}}$ is for $x\le p$:

$$(AG_1 - pG_2)F_1 + xF_2\frac{AG_1 - pG_2}{A} = \frac{(AF_1 + xF_2)(AG_1 - pG_2)}{A}.$$

Next let us consider the case $p \le x$. If the same computation is repeated but instead of $x = 0$ we look at $p = 0$, that is we compute the coefficients of G_1 and pG_2. Putting $p = 0$ in $\frac{e^{-\frac{(x-pe^{-t})^2}{(1-e^{-2t})}}}{\sqrt{\pi}(1-e^{-2t})}$ gives $\frac{e^{-\frac{x^2}{(1-e^{-2t})}}}{\sqrt{\pi}(1-e^{-2t})}$. Its Laplace transform is $AF_1 - xF_2$, carrying through similar computation as was done for the coefficient of xF_2 we get for the coefficient for $pG_2\frac{AF_1 - xF_2}{A}$. Thus the Laplace transform of $\frac{e^{-\frac{(x-pe^{-t})^2}{(1-e^{-2t})}}}{\sqrt{\pi}(1-e^{-2t})}$ when $p \le x$ is: $\frac{(AF_1 + xF_2)(AG_1 - pG_2)}{A}$. Hence the Laplace transform of $\frac{e^{-\frac{(x-pe^{-t})^2}{(1-e^{-2t})}}}{\sqrt{\pi}(1-e^{-2t})}$ is:

$$?\left(\frac{e^{-\frac{(x-pe^{-t})^2}{(1-e^{-2t})}}}{\sqrt{\pi}(1-e^{-2t})}\right) = \begin{cases} \frac{(AF_1 + xF_2)(AG_1 - pG_2)}{A} & \text{if } p \le x \\ \frac{(AF_1 + xF_2)(AG_1 - pG_2)}{A} & \text{if } x \le p. \end{cases} \tag{3.5}$$

4 Level Crossing Probabilites

Let the random variable FC or $FC(x)$ (first crossing) be the smallest possible value of t such that $X_t = x$ given $X_0 = p$. Let $\varphi(t, p, x)$ be the distribution of FC, clearly: $\varphi(t, p, x) * P(t, x, x) = P(t, p, x)$.

That is: $\varphi(t, p, x) * \frac{e^{-\frac{(x-pe^{-t})^2}{(1-e^{-2t})}}}{\sqrt{\pi}(1-e^{-2t})} = \frac{e^{-\frac{(x-pe^{-t})^2}{(1-e^{-2t})}}}{\sqrt{\pi}(1-e^{-2t})}$.

Now the probability that X_t stays below x is: $P\left(\sup_{0\le r\le t} X_r\right) = 1 - \int_0^t \varphi(r)dr$.

Let us denote the Laplace transform of φ by Ψ, then Ψ for $0 \le p \le x$ using (3.5) can be expressed as

$$\psi = \frac{(AG_1(p) + pG_2(p))(AF_1(x) + xF_2(x))}{(AG_1(x) + xG_2(x))(AF_1(x) + xF_2(x))} \tag{4.1}$$

$$= \frac{AG_1 + pG_2}{AG_1 + xG_2} \tag{4.2}$$

For the special case when $p = 0$, that is when X_t reaches level x subject to the initial condition $X_0 = 0$ is

$$\psi = \frac{A}{AG_1 + xG_2}. \tag{4.3}$$

For this case Bellman and Harris [1] found the following expression:

$$\psi = \frac{\frac{1}{2}\Gamma\left(\frac{s}{2}\right)}{\int_0^\infty e^{-y^{2+2xy}} y^{s-1} dy}. \tag{4.4}$$

For the case $p \ge x \ge 0$:

$$\psi(p,x) = \frac{\dfrac{(AF_1 + xF_2)(AG_1 - pG_2)}{A}}{\dfrac{(AF_1 + xF_2)(AG_1 - xG_2)}{A}} \tag{4.5}$$

$$= \frac{AG_1 - pG_2}{AG_1 - xG_2} \tag{4.6}$$

For the special case $p > 0$, $x = 0$ we have:

$$\psi(p,0) = \frac{AG_1 - pG_2}{A}. \tag{4.7}$$

Using (4.7) it is not difficult to show that (4.2) holds for $p \le x$ and holds for $p \ge x$ as well. Formula (4.7) easily invertable, for

$$\frac{1}{A}=\frac{2\Gamma\left(\frac{s+1}{2}\right)}{\Gamma\left(\frac{s}{2}\right)}=\frac{s\Gamma\left(\frac{s+1}{2}\right)}{\frac{s}{2}\Gamma\left(\frac{s}{2}\right)}=\frac{2\Gamma\left(\frac{s+1}{2}\right)}{\frac{s}{2}\Gamma\left(\frac{s}{2}+1\right)}.$$

Clearly $\dfrac{\Gamma\left(\frac{s+1}{2}\right)}{\Gamma\left(\frac{s}{2}+1\right)}$ is the Laplace transform of $2\cdot\dfrac{e^{-t}}{\sqrt{\pi\left(1-e^{-2t}\right)}}$. Hence (4.7) is the Laplace transform of:

$$2\cdot\frac{d}{dt}\left\{\frac{e^{-\frac{p^2e^{-2t}}{\left(1-e^{-2t}\right)}}}{\sqrt{\pi\left(1-e^{-2t}\right)}}*\frac{e^{-t}}{\sqrt{\pi\left(1-e^{-2t}\right)}}\right\}=2\cdot\frac{d}{dt}\frac{1}{\sqrt{\pi}}\int_{\frac{pe^{-t}}{\sqrt{1-e^{-2t}}}}^{\infty}e^{-z^2}dz=\frac{2pe^{-t}}{\sqrt{\pi}}\frac{e^{-\frac{p^2e^{-2t}}{1-e^{-2t}}}}{\left(1-e^{-2t}\right)^{\frac{3}{2}}}.$$

5 The Equivalence of Bellman-Haris' and our Result

To show that formulas (4.3) and (4.4) are the same, we have to evaluate the integral $\int_0^\infty e^{-y^2+2xy}=e^{x^2}\cdot e^{-(x-y)^2}=e^{x^2}\sum_{n=0}^{\infty}(-1)^n\frac{x^n}{n!}\frac{d^ne^{-y^2}}{dy^n}$.

Substituting this into the integral we get:

$$\int_0^\infty e^{-y^2+2xy}y^{s-1}dy=e^{x^2}\cdot\int_0^\infty e^{(x-y)^2}y^{s-1}dy=e^{x^2}\sum_{n=0}^{\infty}(-1)^n\frac{x^n}{n!}\int_0^\infty\frac{d^ne^{-y^2}}{dy^n}y^{s-1}dy.\tag{5.1}$$

Let us observe that the integrals on the right hand side are the Mellin transfroms of the functions $\dfrac{d^ne^{-y^2}}{dy^n}$. First we compute the Mellin transform of e^{-y^2} which is:

$$\int_0^\infty e^{-y^2}y^{s-1}dy=\frac{1}{2}\int_0^\infty e^{-y}y^{\frac{s}{2}-1}dy=\frac{1}{2}\Gamma\left(\frac{s}{2}\right).$$

Let us denote the Mellin transform of a function f by ? or F. It is not difficult to see that:

$$\begin{aligned}
&M(f')=-(s-1)F(s-1)\\
&M(f'')=-(s-1)(s-2)F(s-2)\\
&\cdots\\
&\cdots\\
&M(f^n)=(-1)^n(s-1)(s-2)\cdots(s-n)F(s-n).
\end{aligned}$$

Hence the Mellin transforms of $e^{-y^2}, \frac{de^{-y^2}}{dy}, \frac{d^2e^{-y^2}}{dy^2}, \frac{d^3e^{-y^2}}{dy^3}\cdots$ are

$$\frac{1}{2}\Gamma\left(\frac{s}{2}\right), -\frac{(s-1)}{2}\Gamma\frac{(s-1)}{2}, \frac{(s-2)(s-1)}{2}\Gamma\frac{(S-2)}{2}, \frac{(s-3)(s-2)(s-1)}{2}\Gamma\frac{(S-3)}{2},$$
$$\frac{(s-4)(s-3)(s-2)(s-1)}{2}\Gamma\frac{(S-4)}{2}, \frac{(s-5)(s-4)(s-3)(s-2)(s-1)}{2}\Gamma\frac{(S-5)}{2},\cdots.$$

Substituing these into (5.1) gives:

$$\begin{aligned}
&e^{x^2}\left\{\frac{1}{2}\Gamma\left(\frac{s}{2}\right)+\frac{x}{1!}\frac{s-1}{2}\Gamma\frac{(s-1)}{2}+\frac{x^2}{2!}\frac{(s-2)(s-1)}{2}\Gamma\frac{(s-2)}{2}\right.\\
&+\frac{x^3}{3!}\frac{(s-3)(s-2)(s-1)}{2}\Gamma\frac{(s-3)}{2}+\frac{x^4}{4!}\frac{(s-4)(s-3)(s-2)(s-1)}{2}\Gamma\frac{(s-4)}{2}\\
&\left.+\frac{x^5}{5!}\frac{(s-5)(s-4)(s-3)(s-2)(s-1)}{2}\Gamma\frac{(s-5)}{2}+\cdots\right\}\\
&=e^{x^2}\left\{\frac{1}{2}\Gamma\left(\frac{s}{2}\right)+\frac{x^2}{2!}\frac{(s-2)(s-1)}{2}\Gamma\frac{(s-2)}{2}\right.\\
&\left.+\frac{x^4}{4!}\frac{(s-4)(s-3)(s-2)(s-1)}{2}\Gamma\frac{(s-4)}{2}+\cdots\right\}\\
&+e^{x^2}\left\{\frac{x}{1!}\frac{s-1}{2}\Gamma\frac{(s-1)}{2}+\frac{x^3}{3!}\frac{(s-3)(s-2)(s-1)}{2}\Gamma\frac{(s-3)}{2}\right.\\
&\left.+\frac{x^5}{5!}\frac{(s-5)(s-4)(s-3)(s-2)(s-1)}{2}\Gamma\frac{(s-5)}{2}+\cdots\right\}\\
&=e^{x^2}\left\{\frac{1}{2}\Gamma\left(\frac{s}{2}\right)+\frac{x^2}{2!}2^0\Gamma(s-1)\Gamma\left(\frac{s}{2}\right)+\frac{x^4}{4!}2^1(s-3)(s-1)\Gamma\left(\frac{s}{2}\right)+\cdots\right\}
\end{aligned}$$

$$+e^{x^2}\left\{\frac{x}{1!}2^0\Gamma\left(\frac{s+1}{2}\right)+\frac{x^3}{3!}2^1(s-2)\Gamma\left(\frac{s+1}{2}\right)\right.$$

$$\left.+\frac{x^5}{5!}2^2(s-4)(s-2)(s-1)\Gamma\left(\frac{s+1}{2}\right)+\cdots\right\}$$

$$=e^{x^2}\frac{1}{2}\Gamma\left(\frac{s}{2}\right)\left\{1-\frac{x^2}{1!}\frac{\frac{1-s}{2}}{\frac{1}{2}}+\frac{x^4}{2!}\frac{\frac{1-s}{2}}{\frac{1}{2}}\frac{\frac{3-s}{2}}{\frac{3}{2}}+\cdots\right\}$$

$$+e^{x^2}\Gamma\left(\frac{s+1}{2}\right)x\left\{1-\frac{x^2}{1!}\frac{1-\frac{s}{2}}{\frac{3}{2}}+\frac{x^4}{2!}\frac{1-\frac{s}{2}}{\frac{3}{2}}\frac{2-\frac{s}{2}}{\frac{5}{2}}+\cdots\right\}$$

$$=e^{x^2}\frac{1}{2}\Gamma\left(\frac{s}{2}\right)F\left(\frac{1-s}{2};\frac{1}{2},-x^2\right)+e^{x^2}\Gamma\left(\frac{s+1}{2}\right)xF\left(1-\frac{s}{2},\frac{3}{2};-x^2\right)$$

$$=\frac{1}{2}\Gamma\left(\frac{s}{2}\right)F\left(\frac{s}{2},\frac{1}{2};-x^2\right)+\Gamma\left(\frac{s+1}{2}\right)xF\left(\frac{1+s}{2},\frac{3}{2};x^2\right).$$

Hence Bellman and Harrises formula becomes:

$$\frac{\frac{1}{2}\Gamma\left(\frac{s}{2}\right)}{\int_0^\infty e^{-y^2+2xy}y^{s-1}dy}=\frac{\frac{1}{2}\Gamma\left(\frac{s}{2}\right)}{\frac{1}{2}\Gamma\left(\frac{s}{2}\right)F\left(\frac{s}{2},\frac{1}{2};x^2\right)+\Gamma\left(\frac{s+1}{2}\right)xF\left(\frac{1+s}{2},\frac{3}{2};x^2\right)}.$$

Diving both the numerator and the denominator of the right hand side into $\Gamma\left(\frac{s+1}{2}\right)$ gives $\frac{A}{AG_1+xG_2}$ and this completes the proof.

Reference

[1] Bellman, R., Harris, T.: Recurence times for the Ehrenfest model. Pacific J. Math. 1. 179-193 (1951)

A Bagging Method using Decision Trees in the Role of Base Classifiers

Kristína Machová, František Barčák, Peter Bednár

Department of Cybernetics and Artificial Intelligence, Technical University, Letná 9, 04200 Košice, Slovakia
Kristina.Machova@tuke.sk
Frantisek.Barcak@accenture.com
Peter.Bednár@tuke.sk

Abstract: This paper describes a set of experiments with bagging – a method, which can improve results of classification algorithms. Our use of this method aims at classification algorithms generating decision trees. Results of performance tests focused on the use of the bagging method on binary decision trees are presented. The minimum number of decision trees, which enables an improvement of the classification performed by the bagging method was found. The tests were carried out using the Reuters 21578 collection of documents as well as documents from an internet portal of TV broadcasting company Markíza.

Keywords: classification algorithms, bagging, binary decision trees, text categorisation, recall and precision

1 Introduction

Nowadays, information and data are stored everywhere, mainly on the Internet. To serve us, information had to be transformed into the form, which people can understand, i.e. into the form of knowledge. This transformation represents a large space for various machine learning algorithms, mainly classification ones. The quality of the transformation heavily depends on the precision of classification algorithms in use.

The precision of classification depends on many aspects. Two of most important aspects are the selection of a classification algorithm for a given task and the selection of a training set. In frame of this paper, we have focused on experiments with training set samples, with the aim to improve the precision of classification results. At present, two various approaches are known. The first approach is based on an idea of making various samples of the training set. A classifier is generated for each of these training set samples by a selected machine learning algorithm. In this way, for *k* variations of the training set we get *k* particular classifiers. The result will be given as a combination of individual particular classifiers. This method is called Bagging in the literature [1]. Another similar method called Boosting [7] performs experiments over training sets as well. This method works with weights of training examples. Higher weights are assigned to incorrectly classified examples. That means, that the importance of these examples is emphasised. After the weights are updated, a new (base) classifier is generated. A final classifier is calculated as a combination of base classifiers. The presented paper focuses on the bagging method in combination with Decision trees in the role of base classifiers.

2 Bagging

Bagging is a method for improving results of machine learning classification algorithms. This method was formulated by Leo Breiman and its name was deduced from the phrase "**b**ootstrap **agg**regat**ing**" [1]. More information about bagging can be found in [3], [4] and [9].

In case of classification into two possible classes, a classification algorithm creates a classifier H: D $\rightarrow$ {-1,1} on the base of a training set of example descriptions (in our case played by a document collection) D. The bagging method creates a sequence of classifiers H_m, m=1,…,M in respect to modifications of the training set. These classifiers are combined into a compound classifier. The prediction of the compound classifier is given as a weighted combination of individual classifier predictions:

$$H(d_i) = sign\left(\sum_{m=1}^{M} \alpha_m H_m(d_i)\right).$$

The meaning of the above given formula can be interpreted as a voting procedure. An example d_i is classified to the class for which the majority of particular classifiers vote. Articles [2] and [6] describe the theory of classifier voting. Parameters α_m, m=1,…,M are determined in such way that more precise classifiers have stronger influence on the final prediction than less precise classifiers. The precision of base classifiers H_m can be only a little bit higher than the precision of a random classification. That is why these classifiers H_m are called weak classifiers.

We experimented with the following bagging algorithm [1]:

A bagging algorithm for multiple classification into several classes.

```
1    Initialisation of the training set $D$
2    for $m = 1, ..., M$
```

2.1 Creation of a new set D_m of the same size $|D|$ by random selection of training examples from the set D (some of examples can be selected repeatedly and some may mot be selected at all).

2.2 Learning of a particular classifier H_m: $D_m \to R$ by a given machine learning algorithm based on the actual training set D_m.

3 Compound classifier H is created as the aggregation of particular classifiers H_m: $m = 1, ...,M$ and an example d_i is classified to the class c_j in accordance with the number of votes obtained from particular classifiers H_m.

$$H(d_i, c_j) = sign\left(\sum_{m=1}^{M} \alpha_m H_m(d_i, c_j)\right)$$

If it is possible to influence the learning procedure performed by the classifier H_m directly, classification error can be minimised also by H_m while keeping parameters α_m constant.

The above described algorithm represents an approach called **base version of bagging**. There are some other strategies called **bagging like strategies** which work with smaller size of the training set of example descriptions. These strategies use a combination of the bagging method and the cross-validation method. The **cross-validation** represents the division of the training set into N subsets of D/N size. One of these subsets is used as the training set and the other subsets play the role of test sets.

In "bagging like strategies" the original training set is divided into N subsets of the same size. Each subset is used to create one classifier – a particular classifier is learned using this subset. A compound classifier is created as the aggregation of particular classifiers. The most known methods are: disjoint partitions, small bags, no replication small bags and disjoint bags. An illustrative example of the subset selection process to form new training subsets from an original one is presented in the rest of this section. The original training set containing sixteen examples is depicted in Figure 1.

A B C D E F G H I J K L M N O P

Figure 1
Original training set *D*

The method of **disjoint partitions** uses random selection to select examples. Each example is selected only once. An example of four new subsets, created from the original training set in Figure 1, is presented in Figure 2. In general, if N subsets are created from the original training set, then each of them contains 1/N part from the original set. Union of particular subsets equals the original training set. For very large original set, partitions enable parallel learning of base classifiers.

A B C D	E F G H	I J K L	M N O P

Figure 2
Disjoint partitions

Classifier H obtained from the aggregation of particular classifiers H_m learnt on disjoint partitions, achieves the best results from all „bagging like strategies".
In the method of **small bags**, each subset is generated independently from the other subsets by random selection of training examples with the possibility to select an example repeatedly. An example can be located in several subsets and/or several times in one subset as well. The training sets illustrated in Figure 3 were obtained from the original set in Figure 1. Union of particular partitions does not guarantee to provide the original training set. Classifiers using the small bags reach the worst results from all „bagging like strategies".

A C H L	B P L P	D I O H	K C F K

Figure 3
Small bags

In the method of **no replication small bags**, each subset is generated independently from the other subsets by random selection of training examples without any replication of examples. An example can occur in one subset, several subsets, or no subset. If it occurs in a subset, then exactly one copy is included in the subset. The training sets illustrated in Figure 4 were obtained from the original set in Figure 1. Union of particular partitions does not guarantee to represent the original training set.

A C H L	O P L N	D I O H	K C F P

Figure 4
No replication small bags

The last method from the above mentioned ones is the method of **disjoint bags**. In this method, size of each subset does not equal $|D|$ but is (slightly) greater. First, examples which occur in the original training set are distributed into subsets. Selection of training examples is performed in the same way as in the method of "disjoint partitions". Then, one or more examples are randomly selected and replicated within each subset. The number of replications has to be the same in each subset. An example of resulting division of training examples is illustrated in Figure 5. Each example from the original set occurs (once or more times) exactly in one subset.

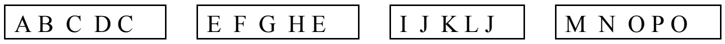


Figure 5
Disjoint bags

Union of particular partitions does not provide the original training set. Classifiers using “disjoint bags" are known to reach the same or sometimes better results as those classifiers using „disjoint partitions“.

3 Text Categorisation

We decided to base our experiments with bagging on the text categorisation task [8]. The aim is to find an approximation of an unknown function $\Phi : D \times C \rightarrow \{true, false\}$ where D is a set of documents and $C = \{c_1, ..., c_{|C|}\}$ is a set of predefined categories. The value of the function Φ for a pair $\langle d_i, c_j\rangle$ is true if the document d_i belongs to the category c_j.. The function $\hat{\Phi} : D \times C \rightarrow \{true, false\}$ which approximates Φ is called a classifier. Definition of the text categorisation task is based on these additional suppositions:

- Categories are only nominal labels and there is no (declarative or procedural) knowledge about their meaning.
- Categorisation is based solely on knowledge extracted from text of the documents.

This definition is a general one and does not require availability of other resources. The constraints may not hold in operational conditions when any kind of knowledge can be used to make the process of categorisation more effective.

Based on a particular application it may be possible to limit the number of categories for which the function Φ has the value true for a given document d_i. If the document d_i can be classified exactly into one class $c_j \in C$, it is the case of the classification into one class and C represents the set of disjoint classes. The case when each document can be classified into an arbitrary number $k = 0, ..., |C|$ of classes from the set C is called multiple classification and C represents the set of overlapping classes.

Binary classification represents a special case when a document can be classified into one of two classes. Classifiers (and algorithms for their creation) for binary classification can be used for multiple classification as well. If classes are independent from each other (i.e. for each pair of classes c_j, c_k, $j \neq k$ holds that the value $\Phi(d_i, c_j)$ is independent from the value $\Phi(d_i, c_k)$), the problem of multiple classification can be decomposed into $|C|$ independent binary classification problems into classes $\{c_i, \overline{c}_i\}$ for $i = 0, ..., |C|$. In this case a classifier for the category c_j stands for the function $\hat{\Phi}_j : D \rightarrow \{true, false\}$, which approximates the unknown function $\Phi_j : D \rightarrow \{true, false\}$.

With respect to the above mentioned decomposition, we used binary decision tree (decision tree performing binary classification) in the role of a base classifier.

4 Classifier Efficiency Evaluation

The evaluation of classifier efficiency can be based on the degree of match between prediction $\hat{\Phi}(d_i, c_j)$ and actual value $\Phi(d_i, c_j)$ calculated over all documents $d_i \in T$ (or $d_i \in V$). Quantitatively it is possible to evaluate the effectiveness in terms of precision and recall (similarly to evaluating methods for information retrieval).

For classification of documents belonging to the class c_j it is possible to define precision π_j as conditional probability $\Pr(\Phi(d_i, c_j) = true \mid \hat{\Phi}(d_i, c_j) = true)$. Similarly, recall ρ_j can be defined as conditional probability $\Pr(\hat{\Phi}(d_i, c_j) = true \mid \Phi(d_i, c_j) = true)$. Probabilities π_j and ρ_j can be estimated from a contingence table Table 1 in the following way:

$$\pi_j = \frac{TP_j}{TP_j + FP_j}, \quad \rho_j = \frac{TP_j}{TP_j + FN_j}$$

where TP_j and TN_j (FP_j and FN_j) are the numbers of correctly (incorrectly) predicted positive and negative examples of the class c_j.

Table 1

Contingence table for category c_j

	$\Phi(d_i, c_j) = true$	$\Phi(d_i, c_j) = false$
$\hat{\Phi}(d_i, c_j) = true$	TP_j	FP_j
$\hat{\Phi}(d_i, c_j) = false$	FN_j	TN_j

Overall precision and recall for all classes can be calculated in two ways. Micro averaging is defined in the following way:

$$\pi^{\mu} = \frac{TP}{TP + FP} = \frac{\sum_{j=1}^{|C|} TP_j}{\sum_{j=1}^{|C|} (TP_j + FP_j)}$$

$$\rho^{\mu} = \frac{TP}{TP + FN} = \frac{\sum_{j=1}^{|C|} TP_j}{\sum_{j=1}^{|C|} (TP_j + FN_j)}$$

while macro averaging is given by the following equations:

$$\pi^{M} = \frac{\sum_{j=1}^{|C|} \pi_j}{|C|} \qquad \rho^{M} = \frac{\sum_{j=1}^{|C|} \rho_j}{|C|}$$

The selection of a particular way of averaging depends on a given task. For example, micro averaging reflects mainly classification of cases belonging to frequently occurring classes while macro averaging is more sensitive to classification of cases from less frequent classes.

Precision and recall can be combined into one measure, for example according to the following formula:

$$F_{\beta} = \frac{(\beta^2 + 1)\pi\rho}{\beta^2\pi + \rho}$$

where β expresses trade off between F_β and π and ρ. Very often it can be seen the use of the function F_1 combining precision and recall using equal weights.

Lacking training data (when it is not possible to select a sufficiently representative test set), it is possible to estimate classification efficiency using cross validation when the set of all examples Ω is divided into k subsets $T_1, ..., T_k$ of the same size. For each subset a classifier $\hat{\Phi}_i$ is constructed using $\Omega - T_i$ as a training set and T_i in the role of the test set. Final estimation can be calculated by averaging results of classifiers $\hat{\Phi}_i$ relevant to all subsets.

5 Experiments

A series of experiments was carried out using binary decision trees as base classifiers. Data from two sources were employed. The first one was the *Reuters-21578*[1] document collection, which comprises Reuter's documents from 1987.

[1] Most experiments were carried out using this document collection, if not given otherwise.

The documents were categorised manually. To experiment, we used a XML version of this collection. The collection consists of 674 categories and contains 24242 terms. The documents were divided into training and test sets – the training set consists of 7770 documents and 3019 documents form the test set. After stemming and stop-words removal, the number of terms was reduced to 19864.

The other document collection, used to perform experiments, was formed by documents from the Internet portal of the Markiza broadcasting company. The documents were classified into 96 categories according to their location on the Internet portal www.markiza.sk. The collection consists of 26785 documents in which 280689 terms can be found. In order to ease experiments, the number of terms was reduced to 70172. This form of the collection was divided into the training and test sets using ratio 2:1. The training set is formed by 17790 documents and the test set by 8995 documents. Documents from this collection are in the Slovak language unlike the first collection, whose documents are written in English.

In order to create decision trees, the famous C4.5 algorithm was used [5]. This algorithm is able to form perfect binary trees over training examples for each decision category. To test the bagging method, weak classifiers (not perfect) are necessary. Therefore, the trees generated by the C4.5 method were subsequently pruned.

5.1 Bagging Efficiency Testing

Results achieved by classifiers, based on the bagging algorithm, were compared with those generated by perfect decision trees. Figure 6 depicts differences between precision of the bagging-based classifier and the precision of the perfect decision tree classifier. Data are shown for each classification class separately (the classes are ordered decreasingly according their frequency).

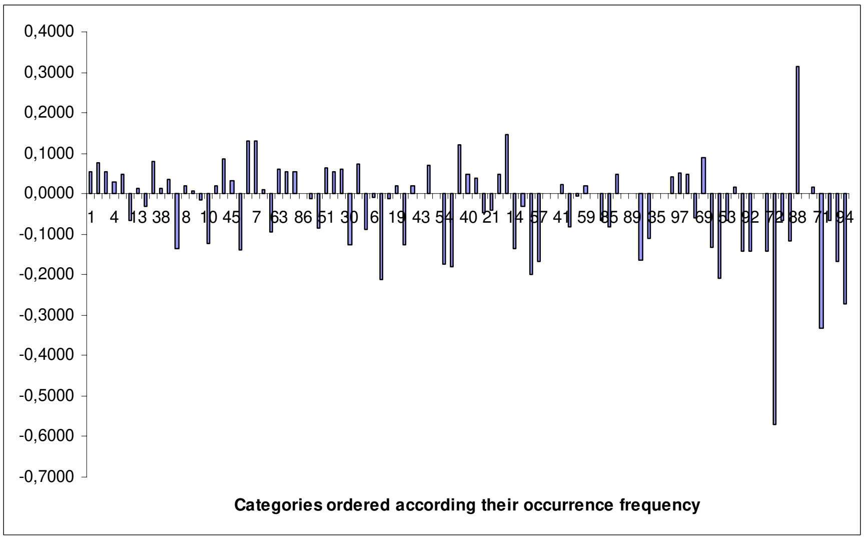


Figure 6
Precision differences between a bagging-based classifier and a perfect decision tree for data from the Reuter's collection

The results can be interpreted in such way that the bagging-based method provides better results than perfect decision trees for more frequent classes. On the other hand, for less frequent classes the results of the perfect decision tree are better.

5.2 Experiments with Different Number of Classifiers

In order to explore dependence of the efficiency of the bagging-based classifier on the number of classifiers, additional experiments were carried out. The number of iterations (i.e. the number of generated binary decision trees) of the bagging algorithm was limited by 200 classifiers. That means, each category was classified by a sum of not more than 200 classifiers. Subsequently, the number of used classifiers was reduced and implications on the classifier efficiency were studied. In order to enable comparison with non-bagging classifier, the efficiency of a perfect binary decision tree was represented on the Figure 7 as a black line.

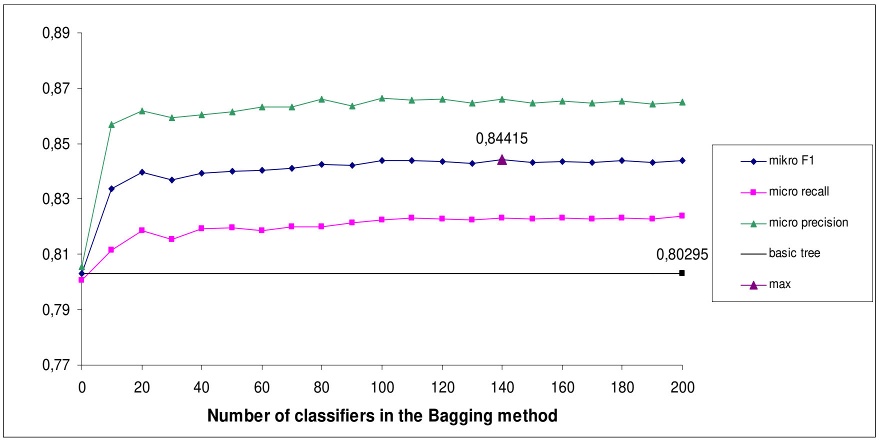


Figure 7

Efficiency differences between the bagging-based classifiers and a perfect decision tree for data from the Reuter's collection

The Figure 7 illustrates that efficiency of the classifiers based on the bagging method does not depend on the quality of particular classifiers (represented by the pruning values), since the values are almost the same for every pruning method. As far as different parameters are concerned, bagging is superior in respect to precision for the number of used classifiers greater than 20. Using 20 or more classifiers, the F1 measure is practically constant. Considering recall, the situation slightly differs. The value of the recall parameter increases with using bigger number of classifiers – with the threshold value 40 classifiers approximately.

Similar experiments were carried out using data from the Internet portal of the Markiza broadcasting company. The results are illustrated on Figure 8. The same parameter setting was used for both the bagging-based classifier and the decision tree classifier.

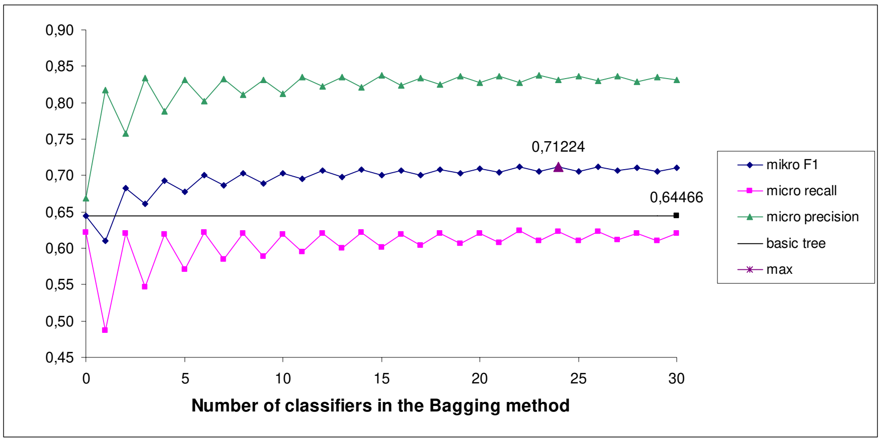


Figure 8
Efficiency differences between the bagging-based classifiers and a perfect decision tree for data from the Markiza collection

The Figure 8 illustrates that as far as different parameters are concerned, the bagging method is superior for the number of classifiers greater than 10 (approximately).

Conclusion

In order to draw a conclusion from our experiments, several statements can be formulated. The bagging method is a suitable mean for increasing efficiency of standard machine learning algorithms.

Considering the same efficiency for a perfect decision tree and bagging-based classifiers, minimum number of classifiers necessary to achieve this efficiency can be found.

As far as disadvantages of bagging are concerned, the loss of simplicity and illustrativeness of this classification scheme can be observed. Increased computational complexity is a bit discouraging as well.

The work presented in this paper was supported by the Slovak Grant Agency of Ministry of Education and Academy of Science of the Slovak Republic within the 1/1060/04 project ”Document classification and annotation for the Semantic web”.

References

[1] Breiman, L.: Bagging predictors. *Technical Report 421, Department of Statistics, University of California at* Berkeley, *1994*

[2] Breiman, L.: Arcing the edge. *Technical report 486, at UC Berkeley, 1996.*

[3] Friedman, J., Springer, L.: <http://www-stat.stanford.edu/people/faculty/friedman.html>

[4] Hastie, T.: <http://www-stat.stanford.edu/%7Ehastie/>Robert

[5] Quinlan, J. R.: Bagging, boosting and C4.5. *In Proc. of the Fourteenth National Conference on Artificial Intelligence, 1996*

[6] Schapire, R., Freund, Y.: Boosting the margin: A new explanation for the efectiveness of voting methods. *The annals of statistics, 26(5):1651-1686, 1998*

[7] Schapire, R. E., Singer, Y.: Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning, 37(3)*, 1999, 297-336

[8] Schapire, R. E., Singer, Y.: BoostTexter: A Boosting – based System for Text Categorization. *Machine Learning, 39(2/3)*, 2000, 135-168

[9] Tibshirani, R.: <http://www-stat.stanford.edu/%7Etibs/>Jerome

Developed Physical Detection-Possibilities of Chemical Agents

Tibor Kovács

Bánki Donát Faculty of Mechanical Engineering, Budapest Tech
Népszínház utca 8, H-1081 Budapest, Hungary
kovacs.tibor@bgk.bmf.hu

Abstract: We can't preclude the possibility of the use of chemical agents by terrorists - I can recall the attempt committed with Sarin in the Tokyo-subway in 1995. For that reason it is necessary to know the chemical situation in the battlefield and in our urban environment as well, for example in the subway. Actually the possible detection principles of toxic-agent-detection-devices are chemical (simple detection devices: e. g. paper detector, detection tubes), biochemical, physical (ion mobility spectroscopy, flame photometry, photoacoustic spectroscopy, infrared or laser remote sensing detection-systems). To control the real time chemical situation it is essential to establish, set up an accurate and rapid reconnaissance. The solution is a monitoring system, which includes developed toxic-agent-detection-devices.

Keywords: Monitoring, real time chemical situation, toxic-agent-detection, principle of physical detection, CAM, AP2C, PAS, remote sensing detection

1 Introduction

Unfortunately, nowadays there are a lot of „chess players" all over the world: the experts suspect that many countries illegally dispose over weapons of mass destruction. According to official data the former Iraqi Dictator used chemical weapons (nerve agent and blister) against his own people. And Saddam didn't shrink from the use of chemical weapons in the war against Iran in 1988. The name of the operation was „Blessed Ramadan" and we can follow the events in *Fig 1*.

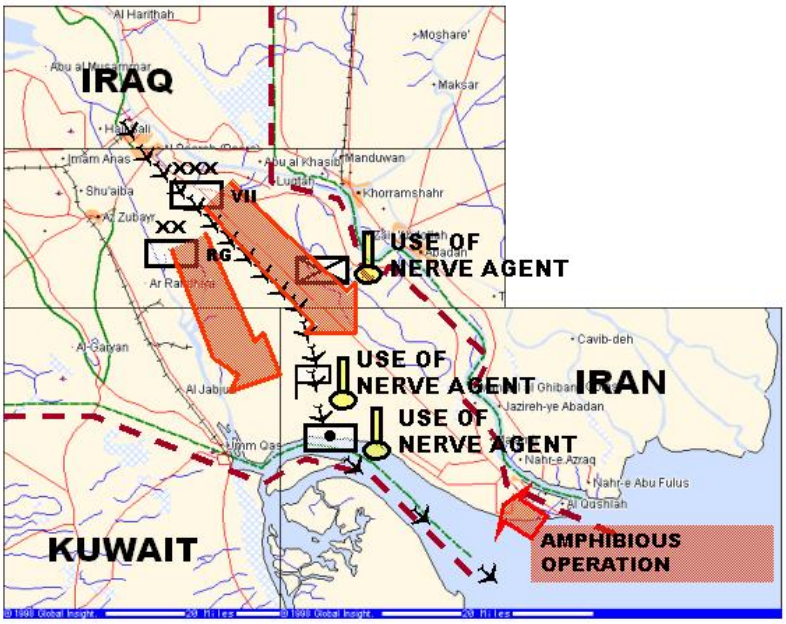


Figure 1

Operation “Blessed Ramadan” (17-21 April, 1988)

On the map the Iraqi 7th Corps and a Division of the Republican Guard are marked (the latter is an elite troop). At first the Iraqi artillery struck some mechanized infantry-troops of the Iranian Armed Forces by nerve agent then the Iraqi troops got moving and the Air Force twice attacked the enemy forces similarly with nerve agent. The operation was finished on the 5th day after an amphibious operation.

As we know during the American attack (2002) Saddam’s weapons of mass destruction disappeared and the expert-group charged by The United Nations in Iraq didn’t find them neither. Likewise we can appoint, unfortunately, there is a real and continuous chemical threat for the Armed Forces and civilian life, as well.

The most important requirements for a developed chemical-detection-device are, that the instrument must be selective, accordingly be able to detect exactly the quality of the toxic agent. It is necessary to be capable to detect simultaneously different toxic agents (e.g. nerve and blister agents together). The concentration must not exceed the 100 mg/m^2 for nerve agents and 10 g/m^2 for blisters. The optimal detection-time is 3-5 s or less. Finally it is very important that the device must be insensitive for non-toxic agents. Naturally, the instrument has to have an output to a personal computer.

2 Ion Mobility Spectroscopy (IMS)

The well known Ion Mobility Spectrometer is the Chemical Agent Monitor. The features of the instrument are the following: it is a hand-held, solder operated, post-attack device for monitoring chemical agents on personnel, equipment or in the field. It has got a Field Alarm Module (FAM), which can provide remote alarm tasks and an automatic switching between nerve and blister modes of operation.

The operational configuration:

- length-width-height: 17-4-7 inches,weight: 1.5 kgs,power supply: single 6 Volts battery (battery life: 6 hrs, continuous use, typical 15 hrs),

- agent concentration sensibility: 0.1 mg/m^2,

- unit cost: $ 6.500

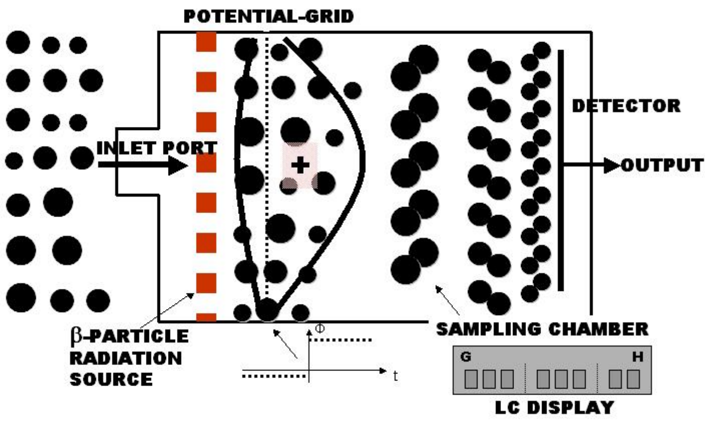


Figure 2

The cross-sectional view of the CAM

The function of CAM (*see: Fig. 2*):

1 The CAM uses a nickel-63 (63Ni) beta-particle radiation source to ionise airborne agent molecules that have been drawn into the sampling chamber by a pump. The resulting ions vary in mass and charge.

2 If the potential grid is negative it collects the positive ions close to it.

3 The form of the ion-cloud is very special due to the effect of pump.

4 If we change to positive the charge of the potential grid the ion-clusters will travel to the detector. The flying time of an ion-cluster depends on its mass (see: Fig. 3).

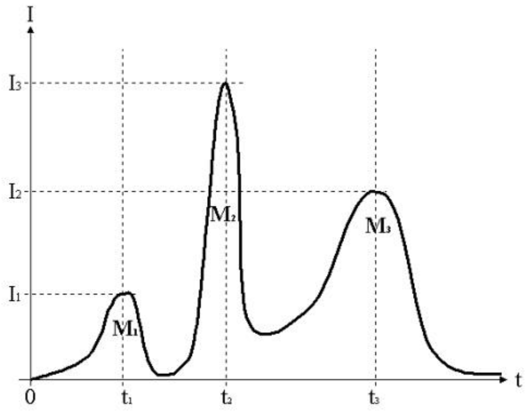


Figure 3
Characteristics of the CAM

5 Flying time data are stored in a ROM, from which the device can determine the type of agent and its relative concentration.

6 A liquid crystal display presents these data as a series of concentration-dependent bars.

7 The agent concentration depends on the wind velocity and other environmental factors, for that reason the numerical display of agent concentration in typical units is impractical.

8 So, the low agent-concentration is marked by 1 to 3 bars, a high 4 to 6 bars, and a very high 7 to 8 bars on LCD.

3 Flame Photometry

Another important principle is the flame photometry. A flame of hydrogen is allowed to burn the air-sample after at the colour of the flame is investigated by a photometer. In this way, the presence of phosphorus and sulphur can be demonstrated. E.g. the most important of this type of instruments is the French monitor AP2C (*Appareil Portatif pour le Contrôle Chimique*). The instrument is demonstrated in the *Fig. 4*.

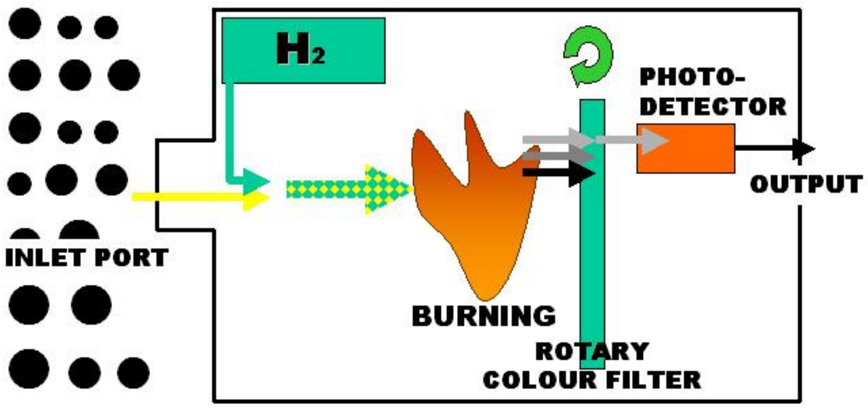


Figure 4

The AP2C (flame photometer)

The operational configuration:

- length-width-height: 16-5-6 inches,
- power supply: single 9…32 Volts battery or rechargeable battery,
- weight: 2.0 kgs,
- single handed operation,
- agent concentration sensibility
  - nerve toxic: 10^{-5} mg/dm^3,
  - blister: $4 \cdot 10^{-3}$ mg/dm^3,
- unit cost: $ 9.500,
- response time < 2 seconds,
- start up time < 1 minute,
- detects all types of mixtures and degraded chemicals,
- used to detect toxic industrial materials (TIMs).

4 Photoacoustic Spectroscopy (PAS)

The origins of Photoacoustic Spectroscopy (PAS) date back to the discovery of the photoacoustic effect by Alexander Graham Bell in 1880.

Bell found that when light was focused onto thin diaphragms, sound was emitted. In latter experiments, Bell studied the sounds produced by the irradiation of various solid samples in a brass cavity sealed with a glass window.

PAS is a non-destructive technique that is applicable to almost all types of samples. It offers minimal or no sample preparation.

PAS can be used for both qualitative and quantitative analysis. In particular, depth profiling experiments are also useful for the characterization of surface-coated and laminar materials and for studies of the diffusion of species into or out of a material.

The phenomenon known as the photoacoustic effect is the emission of sound by an enclosed sample on the absorption of chopped light.

Zero technique measurement, non-destructive invasive analysis and easy-to-use quality are the main performances of the techniques leading today to a large application of photoacoustic detections.

When a gas is irritated with light, it absorbs some of the incident light energy, proportional to the concentration of the gas. The absorbed light energy is immediately released as heat and this causes the pressure to rise. When the incident light is modulated at a given frequency, the pressure-increase is periodic at the modulation frequency. Pressure waves, or sound waves, as they commonly known, are easily measured with a microphone. They are audible if their frequency is between 20 Hz and 20.000 Hz (*see: Fig. 5*).

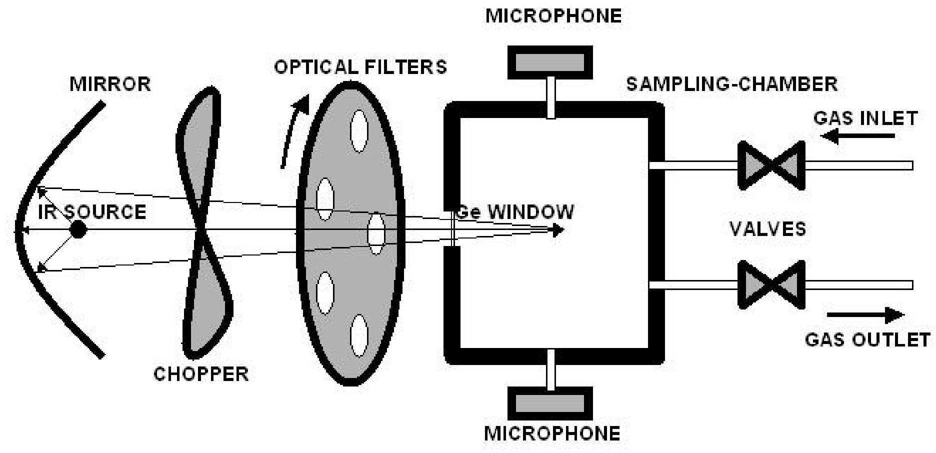


Figure 5

The set-up of the Photoacoustic Spectrometer

The intensity of the sound emitted depends on a number of factors: the nature and concentration of the substance and the intensity of the incident light.

The selectivity which can be achieved in spectroscopy is due to the fact that substances absorb light of specific wavelengths which are characteristic of that substance.

The ability to detect and monitor chemical agents in the event of an attack or after an industrial chemical accident is vital for the efficient use of military and civil defence resources. Systems offering such detection capabilities need to be reliable, accurate and easy-to-use. All principles mentioned are able to fulfil the hardest conditions of the chemical detection.

5 Infrared or Laser Remote Sensing Detection-Systems

(M21 Remote Sensing Chemical Agent Alarm)

The need for detecting a chemical agent cloud from a clean area has been evident for a long time. Advance information of a chemical agent vapour hazard will allow the commander to chose an alternate route or take protective posture just before entering the contaminated area.

The research for remotely detecting chemical agent vapours was first initiated in the late 1950's using infrared technology.

The M21 Automatic Chemical Agent Alarm provides the Army with the first ever capability of detecting chemical agent vapour clouds at a distance.

The M21 Alarm detects nerve and blister agent vapour clouds at line-of-sight distances out to 5 km (*see: Fig. 6*):

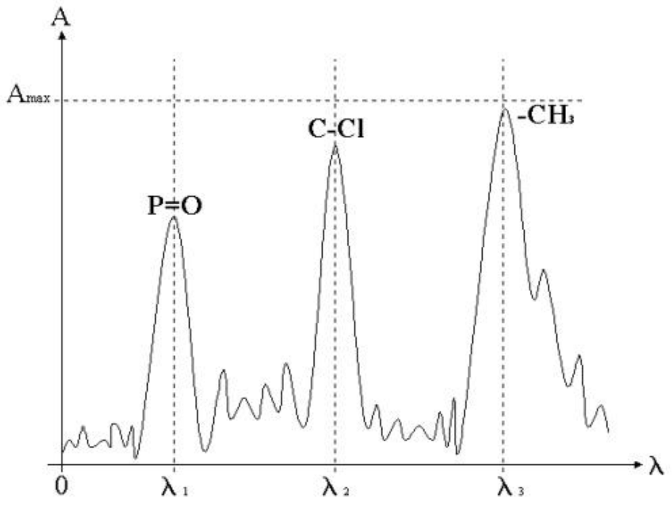


Figure 6
Characteristics of a Remote Sensing Detector

The M21 Alarm operates in the 8-12 micron region of the infrared spectrum. A Michaelson interferometer, the heart of the M21 Alarm, collects absorption or emission spectra from the chemical agent cloud and compares it to the background spectra, so it is a passive infrared device.

The next generation remote detector will provide detection on-the-move and scanning in 360 degrees.

[Lightweight Standoff Chemical Agent Detector (LSCAD)]

The LSCAD is a small, fully-automatic, standoff chemical agent detector.

It is a passive infrared detection system that detects the presence or absence of chemical warfare agents in the 800 to 1200 wave number region of the electromagnetic spectrum by monitoring the ambient background infrared radiation. The signal processing hardware discriminates between the chemical targets and the other non-toxic species in a complex battlefield environment.

The unit is capable of on-the-move, real-time detection from either aerial or surface platforms.

The unit will detect and alarm to a chemical agent cloud up to 5 kms away.

The detector also provides chemical identification information and data for activation of countermeasures to avoid contamination.

The LSCAD is equipped for visual and audible alarm and can display the agent class and relative position. This information is available locally and transmission to battlefield information networks.

LSCAD also has the capability to indicate an all-clear condition.

Conclusions

On the basis of the principles of operation and the most important characteristics of the developed instruments used for the detection of toxic agents we can state:

- the ion mobility spectrometers (and ion mobility spectroscopy) are the best in the field of the chemical reconnaissance,
- the flame photometers are indispensable during the control of decontamination,
- the photoacoustic spectrometers are outstanding in monitoring systems and finally,
- we can use remote sensing detectors if the concentration of toxic agents are relatively high.

References

[1] R. Pellérdi: The Necessities and Possibilities of the Development of the Hungarian NBC Monitoring System, Symposium, on the Most Developed Detection-Possibilities of Toxic Agents, Budapest, Hungary, 1999

[2] T. Kovacs: The Possibilities of Detection of Chemical Agents by the Most Developed Instruments, Symposium, on the Most Developed Detection-Possibilities of Toxic Agents, Budapest, Hungary, 1999

[3] F. Enguehard and L. Bertrand: Effects of optical penetration and laser pulse duration on laser generated longitudinal acoustic waves, J. Appl. Phys., Vol. 82, No. 4, August 1997

[4] Hénault, A. Cournoyer, F. Enguehard, L. Bertrand: A study of dynamic thermal expansion using a laser-generated 1-d-model, Progress in natural science, Supp. to Vol. 6, December 1996

[5] H. Marchand, A. Cournoyer, F. Enguehard, L. Bertrand: Phase optimization for quantitative analysis using phase Fourier transform photoacoustic spectroscopy, Opt. Eng., Vol. 36, No. 2, February 1999

[6] T. Kovacs, L. Bertrand: The Different Possibilities of Detection by Photoacoustic Techniques, Hungarian Military Science, No. 4, 2003, pp. 103-109