7. ročník Žilina 1.- 3. júla 2015
Otvorený softvér vo vzdelávaní, výskume a v IT riešeniach
ossconf.soit.sk
OSS Conf

OSSConf 2014: 1–6

# REQUIRED VALUE CLASSIFICATION USING KOHONEN NEURAL NETWORK

MICHAL CHOVANEC

**Abstract.** In this paper we describe situations classification using Kohonen neural network. Results are demonstrated on line following robot, where different curves are classificated. Each classificated situation have corresponding required value output, which go into low level control process. Network weights estimating is processed in real time.

**Key words and phrases.** kohonen neural network, learning controll, iterative learning, PID controll, mobile robot.

**Klasifikácia žiadanej hodnoty Kohonenovou neurónovou sieťou**

**Abstrakt.** V článku je popísaná klasifikácia situacií Kohonenovou neurónovou sieťou. Výsledky sú demonštrované na robotovi, sledujúceho čiaru, kde sú klasifikované rôzne typi zákrut. Každá situácia má zodpovedajúci výstup, ktorý je vstupom do nižšej úrovne riadenia.

**Keywords.** kohonenová neurónová sieť, inteligentné učenie, iteraivne učenie, PID riadenie, mobilný robot.

## Introduction

Is well know, iterative learning control can handle any repetitive control process well [1], [2]. Most of applications we can found in manipulators control in industry.

Some problems, have iterative (or semiterative) character, but more then one required output sequence is required. Consider problem of line following robot : to control robot speed, is necessary to know curve shape. If we can compute associative memory, where input is line shape and output is required speed, we can handle this problem.

## 1. Problem formalisation

Consider two wheel (two motors) robot with differential drive [3]. For motors control signals we can write

$$\begin{aligned} r(n) &= v(n) - d(n) \\ l(n) &= v(n) + d(n) \end{aligned} \tag{1}$$

Where
$r(n)$ is right motor control output
$l(n)$ is left motor control output
$v(n)$ is common control input
$d(n)$ is difference control input

$d(n)$ can be computed well using PD controller [5]. Our goal is to properly estimate $d(n)$ which is speed of robot. Robot motion equation can be written as

$$\begin{aligned}\theta(n) &= (1+\alpha)\theta(n-1) - \alpha\theta(n-2) + b_0(r(n) - l(n)) \\ \nu(n) &= \alpha\nu(n-1) + (1-\alpha)b_1(r(n) + l(n))\end{aligned} \tag{2}$$

Where
$\theta$ is robot orientation (Yaw angle)
$\nu$ is robot speed
$\alpha$ is robot inertia constant, within $(0, 1)$
$b_0$ and $b_1$ are constants depending on wheel distances, wheel diameters, gear ratio, and maximum speed

If required $\theta$ and $\nu$ are know, we can very well use pure PID control defining error as $e_\theta(n) = \theta_r(n) - \theta(n)$, $e_\nu(n) = \nu_r(n) - \nu(n)$, In our line following problem, is $\nu_r(n)$ unknown, value $\theta_r(n)$ $\theta_r(n)$ readed from line position sensor. In fact, inertia of motors is much smaller than inertia of whole robot, for control $\theta(n)$ we can still use PD controller, and required value let be $\theta_r(n) = 0$, which mean robot is straight on line.

Our goal is to properly estimate $v(n)$ and control robot speed, to give enough time for PD control to setup $\theta(n)$. We are minimizing $e_\theta(n)$ and maximizing $\nu(n)$ (corespoding to $v(n)$).

## 2. Controller design

On figure 1 is block diagram of proposed controller.

Input is line position from sensor $s(n)$. All values are normalised into $\langle -1, 1 \rangle$. Required value of $\theta(n)$ is marked as $r(n)$. In simplification case, can be set to 0. Computed error is input into two PD controllers (PID in general, I-term is equal to zero, because 2 ($\theta$ part) have pole on [1, 0]. If I-term is nonzero, there will be double pole on [1, 0] and system will be unstable).

PD controller outputs are mixed using fuzzy mux, which works as

$$d(n) = s'(n)a(n) + (1 - s'(n))b(n) \tag{3}$$

Where
$a(n)$ is PID A output
$b(n)$ is PID B output

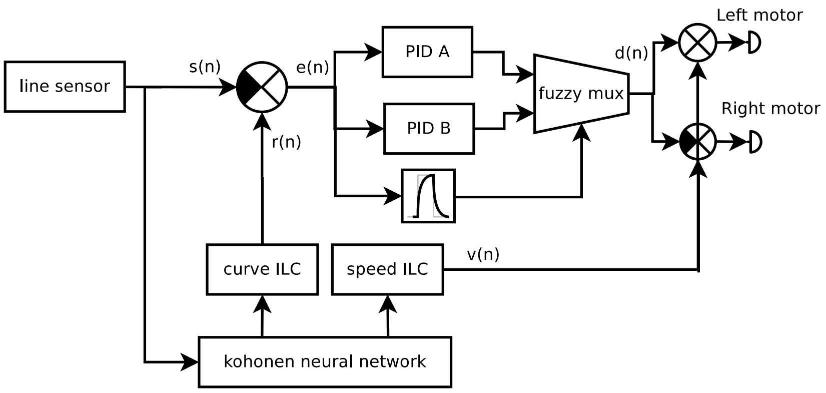

**Figure 1.** Robot controller.

$s'(n)$ is select input, from interval $\langle 0, 1\rangle$

Select output is produced by nonlinear low pass filtered signal of $e(n)$ as

$$s'(n) = \begin{cases} s'(n-1)k + (1-k)|e(n)| & \text{if}|e(n)| < s'(n-1) \\ |e(n)| & else \end{cases} \tag{4}$$

Where $k$ is filter constant, from $(0, 1)$. This filter smooth error values, and rise to huge output value if error immediately rise.

This system properly switching between two controllers, one for straight line one for curved line.

To estimate $v(n)$ is Kohonen neural network used [4] and iterative learning control. Input into neural network is vector of $s(n)$ values (implemented as fifo, in experiments with size M = 16). Mark this vector as $S(n) = [s(n), ..., s(n-M)]$.

Our goal, is to train network to classification vectors $S(n)$. Depending on classes count, we choose corresponding neurons count $N$. In experiments chooses as 16. Which means, 16 different curves shapes can be recognized.

J-th neuron transfer function can be written as

$$y_j(n) = \sum_{i=0}^{M-1} |s(n-i) - w_j(i)| \tag{5}$$

Where $w_j$ are neuron weights, in initialization chosen randomly and modifying during learning process.

Neuron with less $y_q(n)$ will be marked as winning neuron, and weights modification will be processed as

$$w_q(n) = \eta w_q(n-1) + (1-\eta)S(n) \tag{6}$$

Where $\eta$ is learning rate, from $(0, 1)$ close to 1. Weights of neuron are slowly adapted to corresponding pattern. This can be illustrated on figure 2. Input was 2 dimensional, and goal is to find centers (green) of some data set (red). input data set was produced by some Markov process. On figure 3 are show $y_q(n)$ values.

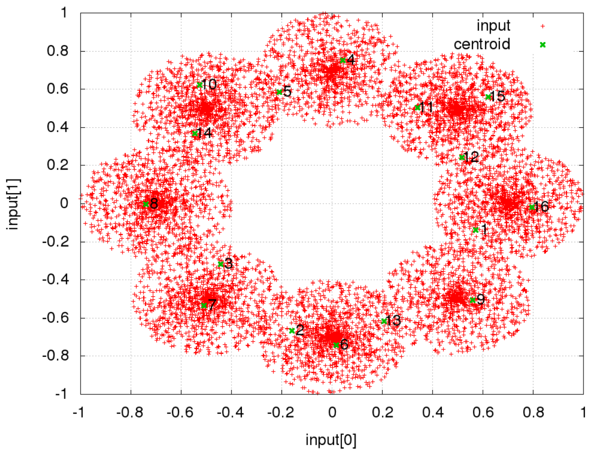

**Figure 2.** Kohonen test.

After found winning neuron, we need to find corresponding $v(n)$, mark this as $v_q(n)$.

$$v_q(n) = v_q(n-1) + 1 - ks_f(n) \tag{7}$$

This means robot speed $v(n)$ on curve type $q$ is rising if $s_f(n)$ is low. Where $s_f(n)$ is low pass filtered $s(n)$ value.

## 3. Experimental results

Robot was learning few loops on short line race with different curves types. Resulting curves types are on figure 4. Each line represents one curve shape. To reduce space, absolute value of $|s(n)|$ as input into Kohonen neural network has been used. We can seen many straight line positions, and few curves.

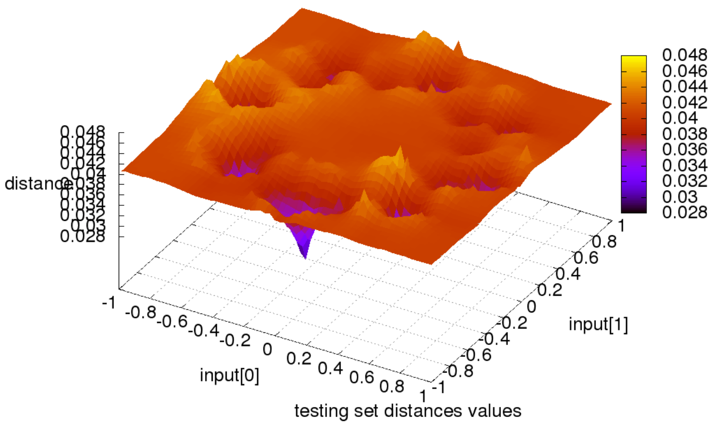

**Figure 3.** Kohonen test distances.

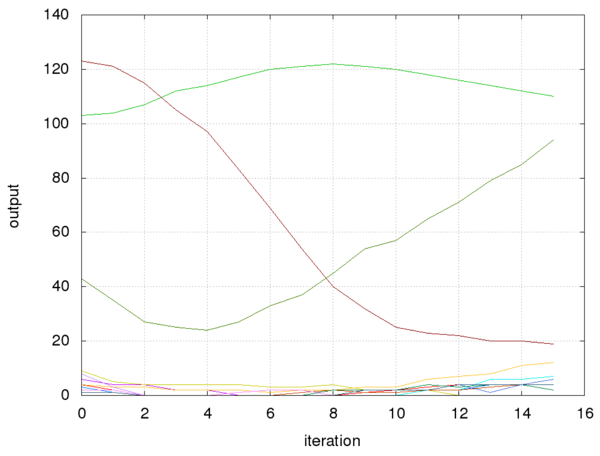

**Figure 4.** Resulting curves types.

## 4. Conclusions

In paper was described learning system for curve shape recognition using Kohonen neural network. Neural network result have corresponding speed output which control robot optional speed. Learning process is working in real time, on 75MHz ARM Cortex M4F microcontroller. Robot is on figure 5. Robot working video can be shown on [6]. More pictures are available on authors blog and sources on github [7].

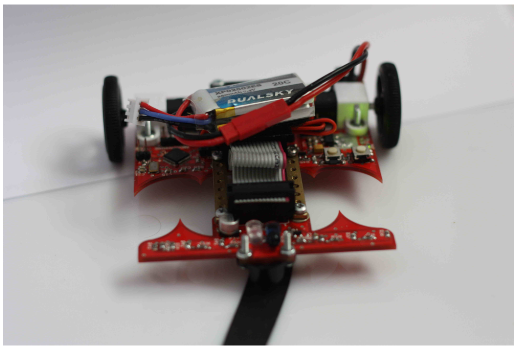

**Figure 5.** Testing robot.

## References

[1] Kevin L. Moore: Iterative learning control `http://inside.mines.edu/~kmoore/survey.pdf`.

[2] Kevin L. Moore: An Introduction to Iterative Learning Control `http://inside.mines.edu/~kmoore/ilc-intro-pdftex03-03.pdf`.

[3] Patrícia N. Guerra: Linear modelling and idendification of a mobile robot with differential drive `http://www.dca.ufrn.br/~adelardo/artigos/ICINCO04b.pdf`.

[4] R. Rojas: Neural Networks, Springer-Verlag, Berlin, 1996, Kohonen Neural Networks `http://page.mi.fu-berlin.de/rojas/neural/chapter/K15.pdf`

[5] Michael Goldfarb, Taweedej Sirithanapipat: The effect of actuator saturation on the performance of PD-controlled servo systems `http://research.vuse.vanderbilt.edu/cim/pubs/journal/35%20-%20Goldfarb%20and%20Sirithanapipat.pdf`.

[6] Michal Chovanec: robot video link `https://www.youtube.com/watch?v=SGaNlbkyQMg`.

[7] Michal Chovanec: authors blog and github `http://aikenshin.blogspot.sk/`. `https://github.com/michalnand/motoko_after_math_linefollower`.

## Contact address

**Ing. Michal Chovanec,** Department of Technical Cybernetics, Faculty of Management Science and Informatics, University of Žilina, Univerzitná 8215/1, 010 26 Žilina, Slovak Republic,
*E-mail address*: `michal.chovanec@yandex.com`, `http://aikenshin.blogspot.sk`