International Conference on Neural Networks and Artificial Intelligence

PROCEEDINGS

12-15 October 1999
Brest, Belarus

Belarus SIG INNS

INNS

Brest 1999

ББК 32.973
УДК 681.3

International Conference on Neural Networks and Artificial Intelligence ICNNAI'99/Proceedings. Edited by Vladimir Golovko. - Brest: BPI, 1999, 224p.

Reviewed by dr. prof. Anatoly Sachenko

Organized by:
· Brest Polytechnic Institute, Department of Computers and Laboratory of Artificial Neural Networks
· Belarus Special Interest Group of International Neural Network Society (SIG INNS)

In cooperation with:
· International Neural Network Society
· Belarusian State University of Informatics and Radioelectronics (Belarus)
· Belarusian Academy of Sciences, Institute of Engineering Cybernetics (Belarus)
· Universidad Politechnica de Valencia (Spain)
· Institute of Computer Information Technologies (Ukraine, Ternopil)

Sponsored by:
· International Association for the promotion of cooperation from the New Independent States of the former Soviet Union (INTAS)
· Ministry of education of Belarus

ISBN 985-6584-05-1

Conference organization

General Conference Chair
Vladislav Fedorov (Belarus)

General Conference Co-chairs
Mihail Golub (Belarus)
Vladimir Golovko (Belarus)

International Program Committee
Pedro Albertos (Spain)
Jesus Pico (Spain)
Herwig Unger (Germany)
Heinrich Werner (Germany)
Wolfram Burgard (Germany)
Hubert Roth (Germany)
Colin Fyfe (United Kingdom)
Darryl Charles (United Kingdom)
Bogdan Gabrys (United Kingdom)
Dominique Luzeaux (France)
Vladimir Shmerko (Poland)
Marco Gori (Italy)
Lucio Grandinetti (Italy)
Alberto Broggi (Italy)
Pasquale Daponte (Italy)
Shuji Yoshizawa (Japan)
Uwe Zimmer (Japan)
Nikola Kasabov (New Zealand)
Kitty Ko (New Zealand)
Alexander Gorban (Russian)
Anatoly Sachenko (Ukraine)
Algirdas Veckys (Lithuania)
Rimvydas Simutis (Lithuania)
Szepesvari Csaba (Hungary)
Valentin Mischenko (Belarus)
Rauf Sadykhov (Belarus)
Vladimir Golenkov (Belarus)
Vecheslav Yarmolik (Belarus)
Anatoly Prihozhy (Belarus)
Alexander Doudkin (Belarus)
Vladimir Ptichkin (Belarus)
Aleksander Krot (Belarus)

Organizing Committee

Chairman
Nikolay Kudinov (Belarus)
Vice Chairman
Jury Savitsky (Belarus)
Committee
Anton Akulich (Belarus)
Victor Brich (Belarus)
Stanislav Derechennik (Belarus)
Vitaly Gladyschuk (Belarus)
Valentin Dimakov (Belarus)
Andrew Dunets (Belarus)
Leonid Mahnist (Belarus)
Vladimir Hveschuk (Belarus)
Aliaksei Klimovich (Belarus)
Larisa Gorbashko (Belarus)
Zhanna Borisyuk (Belarus)
Valery Cylind (Belarus)

International Program Committee Address:
Dr. Vladimir Golovko
Departments of Computers, Brest Polytecnic Institute
Moskowskaja 267, 224017, Brest, Republic of Belarus
e-mail: icnnai99@brpi.belpak.brest.by
fax +375 162 422127, tel. +375 162 421081

Preface

Welcome to ICNNAI'99, the first International Conference on Neural Networks and Artificial Intelligence, which is held in Brest in Belarus. It is hosted and organized by Brest Polytechnic Institute, Laboratory of Artificial Neural Networks, in close collaboration with the International Neural Network Society (INNS). Furthermore, it is supported by the International Association for the promotion of cooperation with scientists from the New Independent States (NIS) of the former Soviet Union (INTAS) and Ministry of Education (Belarus). During the organization of the Conference the new Belarus Special Interest Group of INNS (Belarus SIG INNS) was founded.

The conference aims are to present and discuss together with the researchers from various countries research results and their applications in the broad field of neural computation. It is intended to support worldwide exchange of ideas and to foster dialogue between researchers.

The Conference consists of two complementary parts. The first part is the INTAS monitoring day. On that day different INTAS projects will be presented and discussed as well as some examples of cooperation between West Europe & NIS countries.

The second part is devoted to neural networks and artificial intelligence. It will cover fundamental and applied aspects in the broad field of neural computation. After reviewing, the Program Committee has accepted 40 submissions. Each of the accepted papers has been reviewed by at least two referees. I would like to thank the referees for their work, which was the most important step in the selection process.

And also we would like to thank all those who contributed to the organization of this conference, in the first place the INTAS for their sponsorship and for support.

On behalf of the Program Committee, I want to welcome all the participants to the International Conference on Neural Networks and Artificial Intelligence. I hope that this Conference will be very successful and fruitful to all of you and will contribute to further development in the field of Neural Networks and Artificial Intelligence.

Vladimir Golovko

General Co-Chairman ICNNAI'99

Social Dynamics and Self-Organizing

Vladimir Golovko, Zhanna Borisyuk
Brest Polytechnic Institute, Moskowskaij 267, Brest, 224017, Belarus
phone: 375-162-411081, e-mail: cm@brpi.belpak.brest.by

Abstract

A view of dynamics and self-organizing of social systems is presented. Both a separate Homo sapiens and human society are viewed as social systems. The dynamics of the development of such systems is also examined.. .

1. Introduction

The processes of self-organizing are of great importance for the evolution of social systems. Self-organizing is the process of dynamic reconstruction of an organism or a social system on purpose to adapt to the outer environment. Self-organizing on the level of the separate organism is performed by means of training and as a result there is dynamic reconstruction of neural structures of the head brain. Self-organizing on the level of the social system is realized by means of interaction between the system elements and the outer environment; as a result there is dynamic reconstruction of the connections between the organisms. The processes of dynamics and self-organizing are greatly influenced by the connections imposed on the individual from the side of the society (the organization of the society, religion, morals, etc.). The theoretic ground of the process of self-organizing was developed by I. Prigogin [1-3]. According to him the process of self-organizing takes place in nonequilibrium conditions the source of which are different fluctuations. When the fluctuations exceed some critical point (the point of bifurcation) the system begins its development to a new state. It is impossible to predict the further development of the system in the locality area of the point of bifurcation. It is determined by chance factors. When the choice of the direction has been made, chance gives up its seat to the determined way of development. Such a process is typical of physical systems and doesn't always reflect adeqatety the dynamics of social systems. In this paper the dynamics of social systems is examined and the conclusion is made that in the process of development of the social system the bifurcation approach will more give in to the process of smooth transition to a new quality. The process of changing of entropy of social systems is also being analyzed. Self-organizing of the separate organism is deeply examined.

2. Self-organizing of social systems

Let's examine the evolution and self-organizing of higher biological systems on the level of human society. Homo sapiens are considered to be the component of such a system. In comparison with other organisms Homo sapiens has high level of intelligence and mind, which characterizes his ability for cognition and understanding, for purposive and conscious activity. The individuals are situated there on more high stage of differentiation (more distinguishable): each of them has his own level of development, character, experience and history, i.e. the things, which form his individuality. The evolution of higher biological systems depends on the connections, which exist within the system (the organization of the society). It also depends on the level of development and interaction of its components, historical past and the influence of the outer environment. The connections of the system limit the degree of freedom of its elements and reflect the structure of the society. The interaction and development of both the components and the whole system depend on the connections. The entropy of the separate system may alter according to oscillatory law preserving in the globally the tendency to diminution. Chaotic state of the society (crisis, civil war) or hierarchical organization of the society with rigid connections may serve as an example of increase of the entropy of the biological system in the process of its development. Such organization has its roots in the idea of ancient Greek philosopher Aristotel about natural hierarchy and it is characterized by monodirection of the connections (from top to bottom). Here in fact, the connections limit the degree of freedom of the system components and neutralize the process of collective behavior that is

able to reflect the system to more high levels of organization. In physics the crystals in which the molecules are forged by connections are considered to be the analogy of such an organization of the society. The structure of the crystal is inert and in case of isolation such a state may exist for a very long period of time. Social systems are open and this assists their transformation in time on the purpose to adapt to the outer environment. Different kinds of fluctuations may appear in the process of functioning of the biological system because of influences of inner and outer environment. These fluctuations lead to the formation of local areas of nonequilibriumness. This assists partial disadaptation of the system, which it strives to neutralize. By this in the system with flexible connections there is their reconstruction to compensate the nonequilibriumness and the system with rigid connections tries to push out the nonequilibriumness as a foreign body. Let's call the system stable if it is able to neutralize disadaptation with outer and inner world by means of development and self-organizing. The given definition presupposes not only passive adaptation in the frame of local reconstruction of previous connections, but also active reconstruction of the structures of the system. Social systems in which the arrow of development in a certain period of time is directed to more low stages of organization in comparison with the previous state may serve as an example of unstability. The phenomenon of resonance is of great importance in the process of transformation of the fluctuation of the system and as a result of this the area of nonequilibriumness of the system broadens very quickly. A small nonequilibriumness echoes in the whole system and captures more and more of its components. It may play both a constructive role when it assists transformation of the system to more high level of organization and a destructive role in the opposite case. Disadaptation of the system, which is reflected on the components of the system, is considered to be the reason for resonance and if there is a coincidence with the inner disadaptation of individuals the chain reaction appears. In the process of broadening of the area of system nonequilibriumness there is a competition between fluctuations and connections of the system, which long for their neutralization. According to Gegel's dialectics the process of unity and struggle of the opposites takes place and it is considered to be the highest principle of development. The bigger the rigidness of the connections the more cruel the competition. As a result if the fluctuations capture a sufficient part of the system the area of indefiniteness appears (the surroundings of the point of bifurcation) which is characterized by the indefiniteness of the choice of further way of the development of the system [3]. In this area the system may begin its development on different trajectories. So it can come back to the previous structure in a renewed state, it can also transfer to a new structure by means of self-organizing (of more high or low level) or it can fall into an unstable chaotic state till it meets with a new point of bifurcation. The system may stay in the area of indefiniteness for a long period of time. In such a state it is characterized by the fact that it becomes sensitive to little influences. And by this even an unimportant idea or action of the individual may strengthen itself to hypertrophy quantity and capture the whole system giving birth to the corresponding structure.

By this the idea must adequately reflect the state of the elements of the system, which depends on the reasons of disadaptation, historic experience and on the carrier of the idea. The more rigid the connections of the system the higher the probability of appearance of cataclysms during the process of transition to a new quality and on the contrary. Thus, in the area of indefiniteness the choice of the further route of development is determined in many ways by chance factors, for example, the presence of corresponding leaders that may use resonance for the realization of their ideas. After the direction of the development of the system has been chosen its behavior becomes more determined. The model of interaction of biological system with the outer' environment is shown in fig. 1.6. According to [3] the play of bifurcation's of both the mechanisms of development and self-organizing of the system lies in the basis of the main mechanism of evolution. However probably in the process of development of the mankind as transition from more low to more high forms of organizations. This principle will more and more give in to the process of smooth (without cataclysms) transition to a new quality. Anyway there is a question: how did the mankind manage to escape from Permanent chaos in the globally ? Perhaps it happened owing to the ability for training and development, historic experience, instinct of self-preservation of the individuals and also owing to the universal values: religion, morals, etc. It is also possible to suppose that the evolutionary arrow of time is laid in the mankind. It is considered to be the transition to more high forms of organization and is closely connected with the evolution of the Universe. All this gives the ground to speak about the fact that there is the instinct of self-preservation in the mankind. Thus the chaotic state reflects the human society inadequately because it contradicts to the biological evolution as the increase of the degree of the organization of the system. At it has been already said above the rigidness of the connections in the system increases the probability of appearance of cataclysms during its evolution as an open system. The potential of

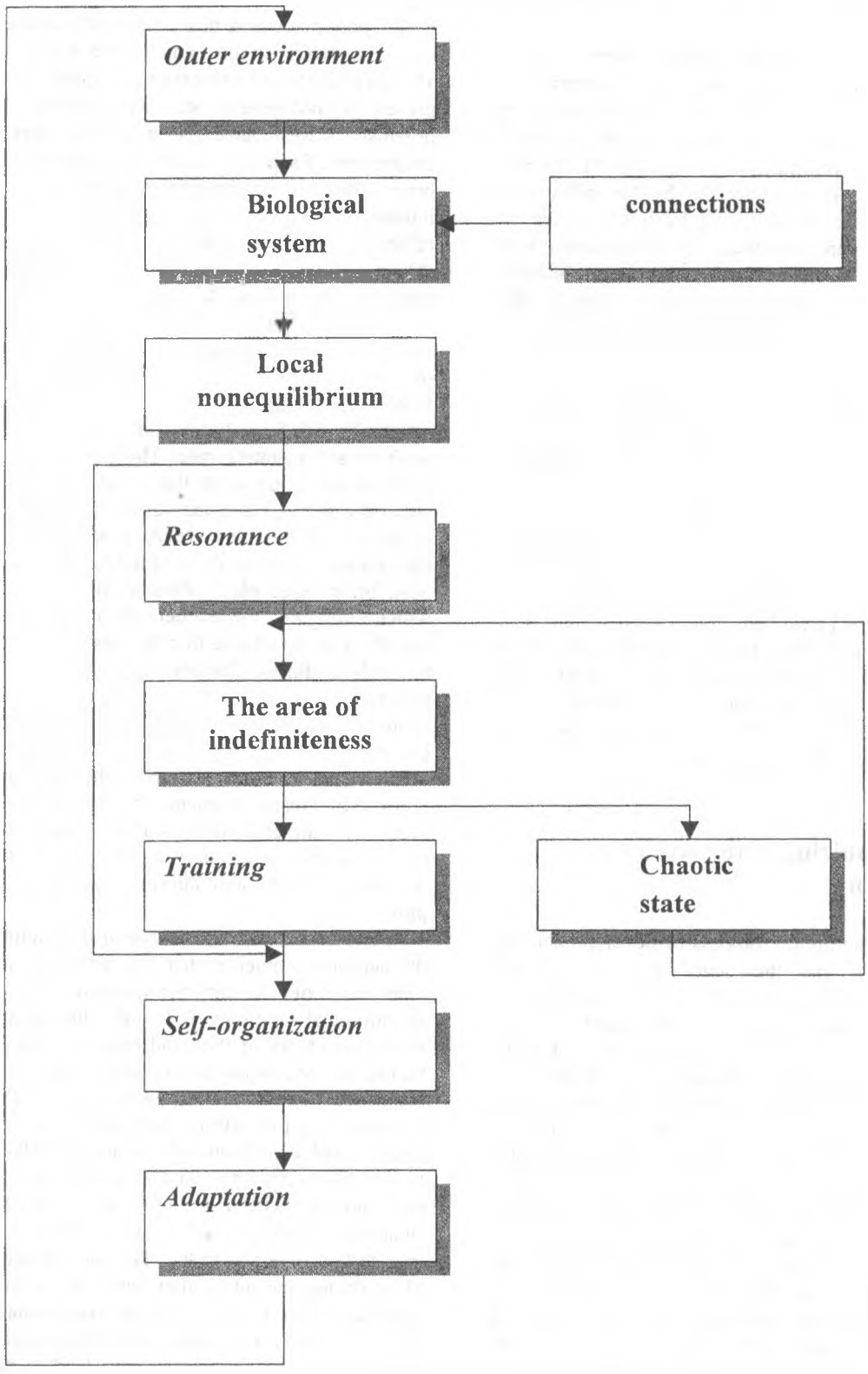


Fig. 1.6. The model of interaction between the biological system and the outer environment.

self-organizing must be laid in the system for its harmonic development.

Let's call a self-organizing system such a system, which is able in the process of interaction of its components with the outer environment to reconstruct its connections and give birth to a new organization on the purpose to develop effectively. Such a system is characterized by flexibility and presence of back connections, by great degree of freedom of its components and by different forms of their collective behavior. In this case the probability of destructive development of the system on the separate time stages diminishes. And by this the resonance assists the smooth reorganization of the system on the purpose to neutralize or disadapt.

Thus the Nonequilibrium State with outer and inner environment is considered to be the source of development of biological systems. Nonequilibrium state is the result of disadaptation of the system and it may generate the process of self-organizing during which the previous structure alters. Reorganization takes place with the help of natural means for self-organizing system (the principle of reorganization is laid in it) and is painful for hierarchical system with rigid connections. The deviation of the entropy of human civilization has an oscillating character and it has a tendency for diminution. In the globally as the historic experience shows human civilization is considered to be the stable system.

3. Self-organizing of the separate organism

Let's examine the process of development and self-organizing on the level of the separate individual who is considered to be the part of the higher biological system. Self-organizing is performed there by means of training and as a result there is a dynamic reconstruction of neural structures of the head brain and consequently the connections of the individual with other components of the system alter. This process constantly takes place and serves for adaptation of the organism to the outer and inner world. From the point of view of absolute knowledge a man stays all the time in the state of chaos (indefiniteness) which characterizes the degree of ignorance. In the process of cognition the size of indefiniteness decreases and in this sense the entropy also diminishes. The source of the development of the individual is represented by nonequilibrium condition with the outer and inner world, which appears as a result of influences from the outer environment. It brings to excitement of the corresponding neural structures of the head brain, which reconstruct themselves in such a way that the individual's behavior could neutralize the arisen nonequilibriumness. Self-organizing of the head brain takes place and this arouses the alteration of the individual's connections in the social system. The phenomenon of resonance influences greatly the process of ontogenesis and self-organizing of the organisms. Resonance is a sharp increase of excitement of separate neural structures of the head brain. This excitement may appear as a result of interaction with the outer environment or as the effect of chemicals. The connections, imposed on the organism (inertionity of synapses, morals, self-preservation instinct, etc.) are considered to be the suppressing factor of resonance. The development of neural activity of the organism depends on the effect of competition between suppressing and stimulating factors. Resonance is one of the devices for adaptation of the organism and it may play both positive and negative role. The positive resonance leads to mobilization of the organism resources to attain the purpose and the negative resonance leads to stresses, depressions, etc. As a result of resonance the dynamic reconstruction of neural networks of the head brain takes place. Positive back connections, which strengthen the excitement of neural activity, are of great importance by this. Resonance may lead to insight during thinking operations when the solution of some problem has been found. For example, the recollections of some attributes connected with the image (name, profession) takes place during its identification. Identification will be realized by means of resonance if the input image is given only the attributes peculiar to him. The same processes take place during the solution of some complex problem and during some other creative processes.

But in this case the degree of the excitement of the neurons is much higher and may continue for a long period of time until the solution of the problem in individual's opinion is found. The inertionity of neural structures of the head brain is considered to be the limiting factor in this process. In the state of resonance even a small influence, if it reflects the reasons of neural activity adequately, may increase sharply and lead to insight. Consequently, chance factors play a great role in high creative processes of the individual because it is necessary that the fluctuation leading to insight was added to the object in a certain time and space. The same process takes place during the interaction between the individual and the system. If there is a negative resonance as it was said above, the excitement of the certain neural structures of the head brain may lead to neuroses, paralysis, schizophrenia, etc.

Affects serve there as a source of excitement. They may be aroused by different associations, for example, by a word. The process of increase of neural activity by means of positive back connections of neural structures may take place as a

result of competition between exciting and suppressing factors (according to Freud-censorship). By this on the analogy of the previous case the organism becomes very sensitive to some influences which are able to lead to catastrophic consequences. Thus, depressive factors may lead to the excitement of neural networks connected with these factors and also it may lead to neutralization of the censorship. Positive back connection strengthens this excitement and resonance takes place. In this state even small influence may arouse the unstability of the organism and this is able to lead to suicide. We see that the chance factors also play great role there. In the state of resonance the behavior of the individual is difficult to predict. The same processes take place as a result of some other physic traumas. So amnesia and paralysis is one of the forms of adaptation of the organism and as a result there is the neutralization of the process of the increase of the excitement. Neutralization is performed with the help of blocking of the corresponding neural structures of the head brain, what arouses these diseases. This process was called "superseding" by S. Freud [4]. The traumas may disappear if there is a back process of disblocking. The disblocking of neural structures may be carried out both under hypnosis and by modeling of stress situations, which have lead to the trauma. According to one of Freud's postulates [4] the thing which has been provocated by the consciousness may be removed by it.

Thus, self-organizing of the separate organism may lead to various consequences. The phenomenon of resonance and accompanying it chance factors play a great role by this.

The alteration of the entropy in the period of active organism's activity has an oscillating character and the entropy has a tendency to decrease. In the passive state when the neurons of head brain begin their liquidating the increase of entropy takes place and this increase attains the maximum value during the transition of the organism to the Quilibrium State with the outer world (the state of death).

References

1. Nicolis G., Prigogine I. Self-organization in Nonequilibrium Systems. - N.Y.: John Wiley and Sons, 1997.
2. Prigogine I. Geoge C. The Second Law as a Selection Principle: The Microscopic Theory of Dissipative Processes in Quantum Systems. Proccedings of the National Academy of Science, 1983, v.80 p.4590-4594.
3. Prigogine I., Stengers I. Order out of chaos. London: Heinemann, 1984.
4. Freud S. Gesammelte Werke. Stuttgart, 1966.

ϵ-insensitive Unsupervised Learning

Colin Fyfe and Bogdan Gabrys,
Applied Computational Intelligence Research Unit,
Department of Computing and Information Systems,
The University of Paisley,
Scotland.

Abstract

One of the major paradigms for unsupervised learning in Artificial Neural Networks is Hebbian learning. The standard implementations of Hebbian learning are optimal under the assumptions of Gaussian noise in a data set. We derive ϵ-insensitive Hebbian learning based on minimising the least absolute error in a compressed data set and show that the learning rule is equivalent to the Principal Component Analysis (PCA) networks' learning rules under a variety of conditions.

1 Introduction

The basis of many unsupervised learning rules is Hebbian learning which is so called after Donald Hebb [9] who conjectured *"When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."* This paper investigates a novel implementation of Hebbian learning

Neural nets which use Hebbian learning are characterised by making the activation of a unit depend on the sum of the weighted activations which feed into the unit. They use a learning rule for these weights which depends on the strength of the simultaneous activation of the sending and receiving neuron. These conditions are usually modelled as

$$y_i = \sum_j w_{ij} x_j \tag{1}$$

$$\text{and } \Delta w_{ij} = \eta x_j y_i \tag{2}$$

the latter being the Hebbian learning mechanism. Here y_i is the output from neuron i, x_j is the j^{th} input, and w_{ij} is the weight from x_j to y_i. η is known as the learning rate and is usually a small scalar which may change with time. We see that the learning mechanism says that if x_j and y_i fire simultaneously, then the weight of the connection between them will be strengthened in proportion to their strengths of firing.

Substituting (1) into (2), we can write the Hebb learning rule as

$$\Delta w_{ij} = \eta x_j \sum_k w_{ik} x_k = \eta \sum_k w_{ik} x_k x_j \tag{3}$$

It is this last equation which gives Hebbian learning its ability to identify the correlations in a data set.

Now it is well known that the simple Hebbian rule above is unstable in that repeated use causes the weights to increase without bounds. Thus weight change rules which have an inbuilt decay term in them have been developed. Many of the most significant of these have been those developed by Oja and colleagues

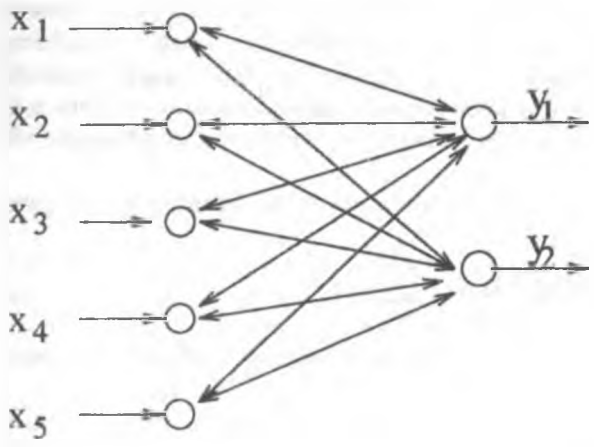


Figure 1: The negative feedback network. Activation transfer is fedforward and summed and returned as inhibition. Adaption is performed by simple Hebbian learning.

[13, 14, 15] which have been shown to not only cause the weights to converge but in particular, to converge so that the network is performing a Principal Component Analysis (PCA) of the data set. It is well known that PCA is the best linear compression of a data set in that it minimises the mean squared error between the compressed data and the original data set. An alternative definition is that PCA provides the linear basis of the data set that captures most variance. This basis is formed from the eigenvectors of the covariance matrix of the data set in order of largest eigenvalues.

In section 2, we describe a negative feedback implementation of Oja's rules and derive a new ϵ-insensitive form of Hebbian learning. In section 3, we experiment with the rule and show that it performs an approximation to Principal Component Analysis and that it is more robust in the presence of shot noise; we also illustrate the effectiveness of the method in an anti-Hebbian rule and in a topology preserving network.

2 The Negative Feedback Network and Cost Functions

Figure 1 shows the network which we have shown to perform a PCA [5] : the data is fed forward from the input neurons (the x-values) to the output neurons. Here the weighted summation of the activations is performed and this is fed back via the same weights and used in the simple Hebbian learning procedure. Consider a network with N dimensional input data and having M output neurons. Then the i^{th} output neuron's activation is given by

$$y_i = \text{act}_i = \sum_{j=1}^{N} w_{ij} x_j \tag{4}$$

with the same notation as before. This firing is fed back through the same weights as inhibition to give

$$x_j(t+1) \leftarrow x_j(t) - \sum_{k=1}^{M} w_{kj} y_k \tag{5}$$

where we have used (t) and $(t+1)$ to differentiate between activation at times t and $t+1$. Now simple Hebbian learning between input and output neurons gives

$$\Delta w_{ij} = \eta_i y_i x_j(t+1) = \eta_i y_i \{x_j(t) - \sum_{l=1}^{M} w_{lj} y_l\} \tag{6}$$

where η_t is the learning rate at time t. This network has previously been shown in [18, 19, 5] to perform a Principal Component Analysis (PCA). In [5], we showed that we may have different weights for feeding back activation from those used for feeding forward activation which accords better with biological neurons in that synapses are one-directional. This network actually only finds the subspace spanned by the Principal Components; we can find the actual Principal Components by introducing some asymmetry into the network [6].

We have previously [8] introduced nonlinearity after the feedback operation in the learning rule to give

$$\Delta w_{ij} = \eta_t f(y_i) x_j(t+1) = \eta_t f(y_i)\{x_j(t) - \sum_{l=1}^{M} w_{lj} y_l\}$$

and related the resulting network to Exploratory Projection Pursuit. We will in section 4 be more interested in the addition of nonlinearity before the feedback,

$$y_i = f(\text{act}_i) = f(\sum_{j=1}^{N} w_{ij} x_j) \tag{7}$$

followed by (5) and (6). [10] have shown that learning rule (6) may be derived as an approximation to gradient descent on the mean squared residuals (5) after feedback.

We can use the residuals (5) after feedback to define a general cost function associated with this network as

$$J = f_1(\mathbf{e}) = f_1(\mathbf{x} - W\mathbf{y}) \tag{8}$$

where in the above $f_1 = ||.||^2$, the (squared) Euclidean norm. It is well known (e.g. [17, 1]) that with this choice of $f_1()$ the cost function is minimised with respect to any set of samples from the data set on the assumption of Gaussian noise on the samples.

It can be shown that, in general (e.g. [17]), the mimimisation of J is equivalent to minimising the negative log probability of the error or residual, e, which may be thought of as the noise in the data set. Thus if we know the probability density function of the residuals, we may use this knowledge to determine the optimal cost function.

It is well known that e.g. speech signals give a data set with kurtotic statistics. An approximation to these density functions is the (one-dimensional) function

$$p(e) = \frac{1}{2+\epsilon} \exp(-|e|_\epsilon) \tag{9}$$

where

$$|e|_\epsilon = \begin{cases} 0 & \forall |e| < \epsilon \\ |e| - \epsilon & \text{otherwise} \end{cases} \tag{10}$$

with ϵ being a small scalar ≥ 0. Using this model of the noise, the optimal $f_1()$ function (to minimise the negative log probability of the error) is the ϵ - insensitive cost function

$$f_1(e) = |e|_\epsilon \tag{11}$$

Therefore when we use this function in the (nonlinear) negative feedback network we get the learning rule

$$\Delta W \propto -\frac{\partial J}{\partial W} = -\frac{\partial f_1(e)}{\partial e}\frac{\partial e}{\partial W}$$

which gives us the learning rules

$$\Delta w_{ij} = \begin{cases} 0 & \text{if } |x_j - \sum_k w_{kj} y_k| < \epsilon \\ \eta . y_i . sign(x_j - \sum_k w_{kj} y_k) = \eta . y . sign(e) & \text{otherwise} \end{cases} \tag{12}$$

where $sign(t) = 1$ if $t > 0$ and $sign(t) = -1$ if $t < 0$.

We see that this is a simplification of the usual Hebb rule using only the sign of the residual rather than the residual itself in the learning rule. We will find that in the linear case (section 3) allowing ϵ to be zero gives generally as accurate results as non-zero values but the data sets in section 4 require non-zero ϵ because of the nature of their innate noise.

2.1 Is this a Hebbian Rule?

The immediate question to be answered is "does this learning rule qualify as a Hebbian learning rule given that the term has a specific connotation in the Artificial Neural Networks literature?". We may consider e.g. covariance learning [12] to be a different form of Hebbian learning but at least it still has the familiar product of inputs and outputs as a central learning term.

The first answer to the question is to compare what Hebb wrote with the equations above. We see that Hebb is quite open about whether the pre-synaptic or the post-synaptic neuron would change (or both) and how the mechanism would work. Indeed it appears that Hebb considered it unlikely that his conjecture could ever be verified (or falsified) since it was so indefinite[4]. Secondly, it is not intended that this method replaces traditional Hebbian learning but that it coexists as a second form of Hebbian learning. This is biologically plausible - *"Just as there are many ways of implementing a Hebbian learning algorithm theoretically, nature may have more than one way of designing a Hebbian synapse"*[2]. Indeed the suggestion that Long Term Potentiation *"seems to involve a heterogeneous family of synaptic changes with dissociable time courses"* seems to favour the coexistance of multiple Hebbian learning mechanisms which learn at different speeds.

We now demonstrate that this new simplified Hebbian rule performs an approximation to Principal Component Analysis in the linear case and finds independent components when we have nonlinear activation functions.

3 ϵ-insensitive Hebbian Learning

In this section we will use the linear neural network

$$
\begin{aligned}
y_i &= \sum_{j=1}^{N} w_{ij} x_j \\
x_j(t+1) &\leftarrow x_j(t) - \sum_{k=1}^{M} w_{kj} y_k \\
\Delta w_{ij} &= \begin{cases} 0 & \text{if } |x_j(t+1)| < \epsilon \\ \eta . y_i . sign(x_j(t+1)) & \text{otherwise} \end{cases}
\end{aligned}
$$

and show that the network converges to the same values to which the more common PCA rules [14, 16] converge.

3.1 Principal Component Analysis

To demonstrate PCA, we use the ϵ-insensitive rules on artificial data. When we use the above learning rules on Gaussian data, we find an approximation to a PCA being performed. The weights shown in Table 1 are from an experiment in which the input data was chosen from zero mean Gaussians in which the first input has the smallest variance, the second the next smallest and so on. Therefore the first Principal Component direction is a vector with zeros everywhere except in the last position which will be a 1 so identifying the filter which minimises the mean square error (which is equivalent to maximising the variance in the projection of the data onto this filter). In our experiment, we have three outputs and five inputs; the weight vector has

2.0749643e-003	6.1149565e-002	**-3.9888122e-001**	**3.6858096e-001**	**-8.3912233e-001**
2.8296234e-002	7.7505112e-003	**8.5505936e-001**	**4.7678573e-001**	**-1.9534208e-001**
5.0981820e-003	-2.5554618e-002	**3.2719504e-001**	**-7.9844131e-001**	**-5.0683308e-001**

Table 1: The subspace spanned by first three principal components is captured after only 5000 iterations. ϵ =0.1

-1.7574163e-002	3.3526187e-002	2.6317708e-002	4.9533961e-002	**1.0068893e+000**
-1.7960927e-002	-2.0583177e-002	2.1191622e-002	**-1.0037621e+000**	6.8503537e-003
-4.1927978e-003	3.6645443e-002	**-9.9189108e-001**	-3.5904231e-002	6.6094374e-002

Table 2: The actual principal components are captured after only 5000 iterations. ϵ =0.5

converged to an orthonormal basis of the principal subspace spanned by the first three principal components: all weights to the inputs with least variance are an order of magnitude smaller than those to the three inputs with most variance. This experiment used 5000 presentations of samples from the data set, $\epsilon = 0.1$ and the learning rate was initially 0.01 and was annealed to 0 during these 5000 iterations.

Just as Oja's Subspace Rule may be transformed into a PCA rule by using deflationary techniques [16] we may find the actual Principal Components by using a deflationary rule with this Hebbian learning. Thus the feedforward rule is as before but feedback and learning occur for each output neuron in turn.

$$\begin{aligned} x_j(t+1) &\leftarrow x_j(t) - w_{kj}y_k \qquad \forall j \\ \Delta w_{kj} &= \eta_t y_k x_j(t+1) \qquad \forall j \end{aligned}$$

for $k = 1, 2....$

Table 2 shows the results when 5 dimensional data of the same type as before was used as input data, the learning rate was 0.1 decreasing to 0 and ϵ was 0.1. These results were taken after only 5000 iterations. The convergence is very fast: a typical set of results from the same data are shown in Table 3 where the simulation was run over only 1000 presentations of the data (We have also used ϵ=0 in this case to demonstrate that the particular value of ϵ is not crucial). So, as might have been expected, minimising the mean square error and minimising the mean absolute error give the same results in the Gaussian case.

We may expect that since the learning rule is insensitive to the magnitude of the input vectors $\mathbf{x}$, the rule is less sensitive to outliers than the usual rule based on mean square error. To test this, we add noise from a uniform distribution in [-10,10] to the last input (that with smallest variance) in 30% of the presentations of the input data. Table 4 shows that the PCA properties of the ϵ-insensitive deflationary network are unaffected by the noise. In comparison, Table 5 shows that the Sanger [16] network responds to this noise (as one would expect).

We note that this need not be a good thing, however in the context of real biological neurons we may wish each individual neuron to ignore high intensity shot noise and so the ϵ-insensitive rule may be optimal. Finally the insertion of a differentiated θ_t term in the calculation of the residual (as in [15]) also causes convergence to the actual Principal Components but was found to be two orders of magnitude slower than the deflationary technique described above.

-5.1776166e-003	4.5376865e-002	-2.3840385e-003	1.2658529e-002	**1.0019231e+000**
-3.6847661e-002	1.0566274e-002	-1.3372074e-001	**9.9144360e-001**	1.6136625e-003
-1.1309247e-002	-8.1029262e-003	**9.9036232e-001**	1.4140185e-001	-1.4256455e-003

Table 3: The actual principal components are almost found after only 1000 iterations. ϵ =0.

2.5339642e-002	-3.9976042e-002	4.7171348e-002	3.3109733e-002	**9.9560379e-001**
1.5587732e-002	1.4625496e-002	9.7532991e-003	**-1.0013772e+000**	3.2922597e-002
-4.8078854e-002	-6.4830431e-002	**9.9738211e-001**	2.1815735e-002	-4.5000940e-002

Table 4: The 30% outliers are ignored by the ϵ-insensitive rule.

6.5436571e-002	-4.2781770e-002	-3.5680891e-002	5.9717521e-002	**-9.9664319e-001**
6.9542470e-002	-1.4830770e-002	1.7929844e-002	**-9.9627047e-001**	-7.1398709e-002
-9.5947374e-001	-1.7583321e-003	-2.1653710e-001	-9.4726964e-002	-1.1765086e-001

Table 5: The standard Sanger rule finds the noise irresistable.

3.2 Anti-Hebbian Learning

Now the ϵ-insensitive rule was derived in the context of the minimisation of a specific function of the residual. It is perhaps of interest to enquire whether similar rules may be used in other forms of Hebbian learning. We investigate this using anti-Hebbian learning.

Földiák [3] has suggested a neural net model which has anti-Hebbian connections between the output neurons.

The equations which define its dynamical behaviour are

$$y_i = x_i + \sum_{j=1}^{N} w_{ij} y_j$$

In matrix terms, we have

$$\begin{aligned} \mathbf{y} &= \mathbf{x} + \mathbf{W}\mathbf{y} \\ \text{And so, } \mathbf{y} &= (\mathbf{I} - \mathbf{W})^{-1}\mathbf{x} \end{aligned}$$

He shows that, after training with the familiar anti-Hebbian rule,

$$\Delta w_{ij} = -\eta y_i y_j \text{ for } i \neq j$$

the outputs, $\mathbf{y}$ are decorrelated.

Now the matrix W must be symmetric and has only non-zero non-diagonal terms i.e. if we consider only a two input, two output net,

$$W = \begin{pmatrix} 0 & w \\ w & 0 \end{pmatrix} \tag{13}$$

However the ϵ-insensitive anti-Hebbian rule is non-symmetrical since, if w_{ij} is the weight from y_i to y_j, we have

$$\begin{aligned} \Delta w_{ij} &= -\eta y_j \, sign(y_i) \text{ if } |y_i| > \epsilon \\ \Delta w_{ij} &= 0 \qquad \text{otherwise} \end{aligned} \tag{14}$$

To test the method, we generated 2 dimensional input vectors, $\mathbf{x}$, where each element is drawn independently from N(0,1) and then added another independently drawn sample from N(0,1) to both elements. This gives a data set with sample covariance matrix (10000 samples) of

$$\begin{pmatrix} 1.9747 & 0.9948 \\ 0.9948 & 1.9806 \end{pmatrix} \tag{15}$$

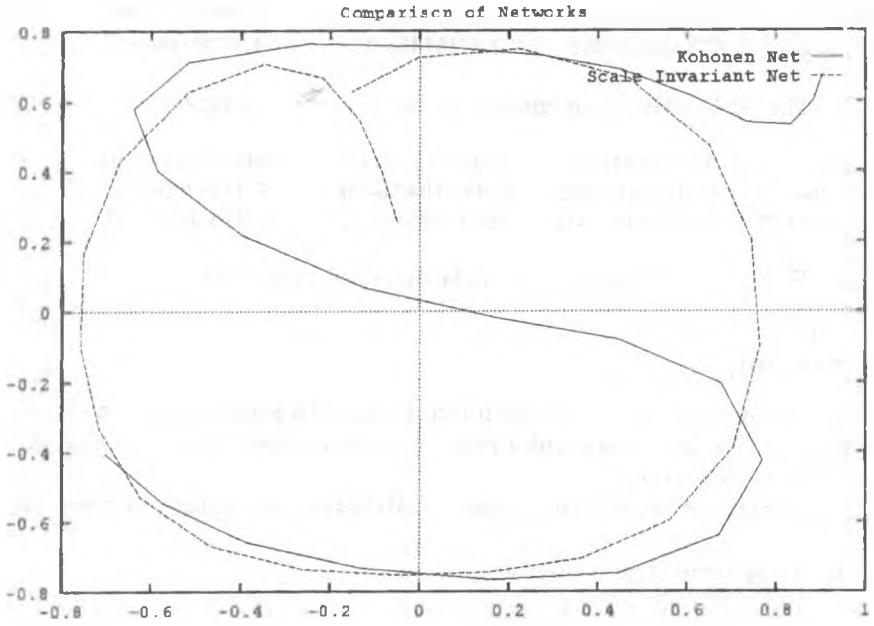


Figure 2: The Kohonen mapping and the Scale Invariant mapping on two dimensional uniform distribution.

The covariance matrix of the outputs, $\mathbf{y}$, (over the 10000 samples) from the network trained using the ϵ-insensitive learning rule is

$$\begin{pmatrix} 1.8881 & -0.0079 \\ -0.0079 & 1.1798 \end{pmatrix} \tag{16}$$

We see that the outputs are almost decorrelated. It is interesting to note that

- the asymmetrical learning rules have resulted in non-equal variances on the outputs.
- but the covariance (off-diagonal) terms are equal.

It is our finding that the outputs are always decorrelated but the final values on the diagonals (the variances) are impossible to predict and seem to depend on the actual values seen in training, the initial conditions etc..

A feedforward decorrelating network, $\mathbf{y} = (I + W)\mathbf{x}$, may also be created with the ϵ-insensitive anti-Hebbian rule with similar resuts.

3.3 Topology Preserving Maps

We have also used the negative feedback network to create topology preserving maps [7] with somewhat different properties from Kohonen's SOM [11]. The feedforward stage is as before but now we invoke a competition between the output neurons and the neuron with greatest activation wins. The winning neuron, the p^{th}, is deemed to be maximally firing (=1) and all other output neurons are suppressed(=0). Its firing is then fed back through the same weights to the input neurons as inhibition.

$$x_j(t+1) \leftarrow x_j(t) - w_{pj}.1 \text{ for all } j \tag{17}$$

where p is the winning neuron. Now the winning neuron excites those neurons close to it i.e. we have a neighbourhood function $\Lambda(p,j)$ which satisfies $\Lambda(p,j) \leq \Lambda(p,k)$ for all $j,k : \| p-j \| \geq \| p-k \|$ where $\| \cdot \|$ is the Euclidean norm. Typically, we use a Gaussian whose radius is decreased during the course of the simulation. Then simple Hebbian learning gives

$$\Delta w_{ij} = \eta_t \Lambda(p,i).x_j(t+1) \quad (18)$$

$$= \eta_t \Lambda(p,i).(x_j(t) - w_{pj}) \quad (19)$$

The somewhat different properties of this mapping from the Kohonen mapping may be seen in Figure 2: the Kohonen SOM spreads out evenly across the data set while with the scale invariant mapping, each output neuron captures a slice of the inputs of approximately equal magnitude angle. We may now report that if we use the ϵ-insensitive learning rule

$$\Delta w_{ij} = \begin{cases} 0 & \text{if } |x_j(t+1)| < \epsilon \\ \eta.\Lambda(p,i).sign(x_j(t+1)) & \text{otherwise} \end{cases} \quad (20)$$

convergence to the same type of mapping is also achieved. As before, the resultant mapping is found more quickly and is more robust against shot noise.

4 Conclusion

We have derived a slightly different form of Hebbian learning which we have shown capable of performing PCA type learning in a linear network. We have shown that the method may also be used for anti-Hebbian learning and for the creation of topology preserving maps.

Future work will concentrate on analysing the form of noise to be expected in real situations and creating cost functions which are optimal for these situations.

References

[1] C. Bishop. *Neural Networks for Pattern Recognition.* Oxford:Clarendon Press, 1995.

[2] T. H. Brown and S. Chatterji. *The Handbook of Brain Theory and Neural Networks*, chapter Hebbian Synaptic Plasticity. MIT Press, 1995.

[3] P. Földiak. *Models of Sensory Coding.* PhD thesis, University of Cambridge, 1992.

[4] Y. Fregnac. *The Handbook of Brain Theory and Neural Networks*, chapter Hebbian Synaptic Plasticity: Comparative and Developmental Aspects. MIT Press, 1995.

[5] C. Fyfe. Pca properties of interneurons. In *From Neurobiology to Real World Computing, ICANN 93*, pages 183–188, 1993.

[6] C. Fyfe. Introducing asymmetry into interneuron learning. *Neural Computation*, 7(6):1167–1181, 1995.

[7] C. Fyfe. A scale invariant feature map. *Network: Computation in Neural Systems*, 7:269–275, 1996.

[8] C. Fyfe and R. Baddeley. Non-linear data structure extraction using simple hebbian networks. *Biological Cybernetics*, 72(6):533–541, 1995.

[9] D. O. Hebb. *The Organisation of Behaviour.* Wiley, 1949.

[10] Juha Karhunen and Jyrki Joutsensalo. Representation and separation of signals using nonlinear pca type learning. *Neural Networks*, 7(1):113–127, 1994.

[11] Tuevo Kohonen. *Self-Organising Maps.* Springer, 1995.

[12] R. Linsker. From basic network principles to neural architecture. In *Proceedings of National Academy of Sciences*, 1986.

[13] E. Oja. A simplified neuron model as a principal component analyser. *Journal of Mathematical Biology*, 16:267–273, 1982.

[14] E. Oja. Neural networks, principal components and subspaces. *International Journal of Neural Systems*, 1:61–68, 1989.

[15] E. Oja, H. Ogawa, and J. Wangviwattana. Pca in fully parallel neural networks. In Aleksander & Taylor, editor, *Artificial Neural Networks,2*, 1992.

[16] T.D. Sanger. Analysis of the two-dimensional receptive fields learned by the generalized hebbian algorithm in response to random input. *Biological Cybernetics*, 1990.

[17] A. J. Smola and B. Scholkopf. A tutorial on support vector regression. Technical Report NC2-TR-1998-030, NeuroCOLT2 Technical Report Series, October 1998.

[18] R. J. Williams. Feature discovery through error-correcting learning. Technical Report 8501, Institute for Cognitive Science, University of California, San Diego, 1985.

[19] Lei Xu. Least mean square error reconstruction principle for self-organizing neural-nets. *Neural Networks*, 6(5):627 - 648, 1993.

Unsupervised Training Algorithm for Recirculation Neural Network

Vladimir Golovko, Vitaly Gladyschuk

Department of Computers, Brest polytechnic institute, Moscowskaja 267, 224017 Brest, Republic of Belarus, ph: +375 162 421081, fax: +375 162 422127, e-mail: cm@brpi.belpak.brest.by

Abstract

Unsupervised learning is the great promise of the future. In such training the network is provided with inputs but not with desired outputs. Unsupervised learning is used for the principal component networks. This paper describes a new method for training of the recirculation networks. Such method is called a sectioning learning. It is characterized by small training time and stability of training.

Keywords. Recirculation neural network, unsupervised training, compression.

1. Introduction

As it is known from statistics[1] the Principal Component Analysis (PCA) is the basic tool to reduce dimension by eliminating redundant variables. Such transformation from the n-dimensional to the m-dimensional vector equals the concatenation of the first m eigenvector of the correlation matrix of the input signals. In this procedure the input space is rotated in such a way that the output values are as uncorrelated as possible and the energy or variances of the data is mainly concentrated in a few first principal components.

The principal component networks (recirculation networks) use the various variants of unsupervised learning. So many papers are based on Hebbian learning [2,3,4]. Such learning rule is the equivalent to the Principal Component Analyses. Other authors [5,6,7] have used the backprogation algorithm or cumulative delta rule. However these algorithms have excessive training times and lack of convergence to an acceptable solution. There is no guarantee that the learning will be a success.

This paper describes a new method for training the recirculation network. Such method is characterized by small training times and stability of training. Various numerical experiments are used to illustrate the potential of the suggested method.

2. Architecture

Recirculation networks are characterized both feed-forward $y = f(x)$ and feed-back $\bar{x} = f(y)$ transformation of the patterns. Such networks are used for compression (feed-forward transformation) and decompression (feed-back transformation) of the data. The architecture of the recirculation neural network is shown on Fig.1. It consists of three layers. The input units receive data from outside and distribute these data to hidden units. The hidden units perform the compression of the input data X:

$$Y = F(WX) \tag{1}$$

The units in the output layer are meant for decompression of the data of the hidden layer:

$$\bar{X} = F(W'Y), \tag{2}$$

where W is the weight matrix; X is the input vector; Y is the output vector of the hidden layer; $\bar{X}$ is the decompressed vector; F is the activation function.

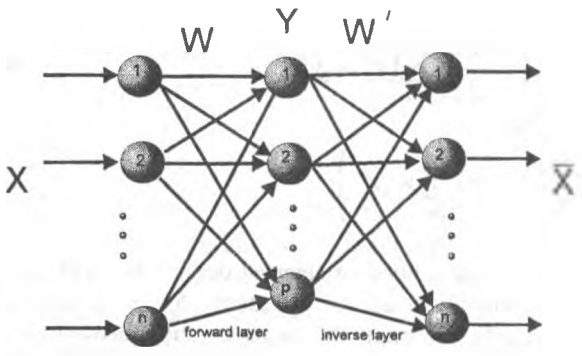


Figure 1.

One can see in Fig.1, that recirculation network has two weight layers, called the forward and inverse layers. The purpose of the learning is to reduce the error between the decompressed vector and the input vector.

3. Background and motivation

The proposed method involves two separate phases of the training. At the first phase the definition of the weight matrix W' and principal components $\bar{Y}$ for minimization of the total square error between the decompressed data and the original data set is performed. For this purpose it is necessary to minimize the following equation:

$$|X - W'\bar{Y}| \to \min \tag{3}$$

It is equivalent to the minimization

$$E_1 = \frac{1}{2}\sum_{k=1}^{L}\sum_{i=1}^{n}(\bar{x}_i^k - x_i^k)^2, \tag{4}$$

where L is the quantity of the input data.
By this

$$\bar{x}_i = F\left(\sum_{j=1}^{p} w'_{ji} y_j\right), \tag{5}$$

where p is the dimension of the hidden layer and $p < n$. For the minimization of the equation (5) the method of steepest descent can be used. As a result the weights w'_{ij} of inverse layer and principal components $\bar{y}_j$ are defined.

At the second phase the weight matrix W of the forward layer is determined. As the target outputs the vector $\bar{Y}$ is used, which has been obtained previously. Then the aim of the training at the second phase is to minimize the following expression:

$$E_2' = \frac{1}{2}\sum_{k=1}^{L}\sum_{i=1}^{n}(y_i^k - \bar{y}_i^k)^2, \tag{6}$$

where

$$y_j = F\left(\sum_{i=1}^{n} w_{ij} x_i\right) \tag{7}$$

The method of steepest descent we will use for minimization of the equation (6). Such approach permits to train the inverse and forward layers separately. Let's consider the training rules for linear and nonlinear networks.

4. Linear network

Such networks use a linear activation function and perform the linear compression of a data set. Then the outputs of the forward layer are as follows

$$y_j = \sum_i w_{ij} x_i, \tag{8}$$

where $j = \overline{1, p}$.
The outputs of the inverse layer are given by

$$\bar{x}_i = \sum_{j=1}^{p} w'_{ji} y_j, \tag{9}$$

where $i = \overline{1, n}$.
The learning problem at the first phase can be formulated as: how do we compute $\Delta w'_{ij}(t)$ and $\Delta y_j(t)$ in order to minimize the total mean-square error between the decompressed data $\bar{X}$ and the original data set X (equation 4)? Let's examine the definition of the vector $\bar{Y}$ for the minimization of the equation (4). For this purpose the gradient descent method is used. Then the learning rule is

$$y_j(t+1) = y_j(t) - \alpha_j(t)\frac{\partial E}{\partial y_j(t)}, \tag{10}$$

where $\alpha_j(t)$ is the adaptive training rate for neuron j of the hidden layer.
The error E is defined by the expression:

$$E = \frac{1}{2}\sum_{i=1}^{n}(\bar{x}_i - x_i)^2 \tag{11}$$

Then the error derivate is

$$\gamma_j = \frac{\partial E}{\partial y_j} = \frac{\partial E}{\partial x_i}\frac{\partial \bar{x}_i}{\partial y_j} = \sum_i w'_{ji}(\bar{x}_i - x_i) \tag{12}$$

Theorem 1. The adaptive training rate $\alpha_j(t)$ is defined by the following expression:

$$\alpha_j(t) = \frac{1}{\sum_i (w'_{ji}(t))^2} \tag{13}$$

Proof. In order to compute the training rate $\alpha_j(t)$ we will use the method of the steepest descent. From this follows that

$$\alpha_j(t) = \min\left\{E(y_j(t) - \alpha_j(t)\frac{\partial E}{\partial y_j(t)}\right\}$$

The activation values of element i can be written as:

$$\bar{x}_i(t+1) = w'_{ji}(y_j(t) - \alpha_j \gamma_j) + \sum_{k \neq j} w_{ki} y_k(t)$$

After modifications we have

$$\bar{x}_i(t+1) = \bar{x}_i(t) - \alpha_j \gamma_j w'_{ji}$$

The error function for a pattern

$$E = \frac{1}{2} \sum_{i=1}^{n} \left(\bar{x}_i(t+1) - x_i\right)^2$$

Then the derivate is

$$\frac{\partial E}{\partial \alpha_j} = \sum_i \left(\bar{x}_i(t) - x_i - \alpha_j \gamma_j w'_{ji}\right) * \left(-\gamma_j w'_{ji}\right) = 0 \quad (14)$$

Now we can determine the adaptive rate.

From equation (14) follows that:

$$\alpha_j = \frac{\gamma_j}{\sum_{i=1}^{n} (w'_{ji}(t))^2} \quad (15)$$

It may be noted, that

$$\frac{\partial^2 E}{\partial \alpha_j^2} > 0$$

From this follows that the equation (15) minimizes the error function. Thus the learning rate is adapted to connection weights w'_{ji}.

The learning rule is

$$y_j(t+1) = y_j(t) - \frac{\gamma_j}{\sum_{i=1}^{n} (w'_{ji}(t))^2} \quad (16)$$

Now we have to determine the learning rule for connection weights w'_{ji}. In accordance to the gradient descent method

$$w'_{ji}(t+1) = w'_{ji}(t) - \alpha(t) \frac{\partial E}{\partial w'_{ji}(t)}, \quad (17)$$

where $\alpha(t)$ is adaptive learning rate for inverse layer.

In this case we can get that

$$w'_{ji}(t+1) = w'_{ji}(t) - \alpha(t)(\bar{x}_i - x_i) y_j. \quad (18)$$

The learning rate can be defined by analogy with theorem 1. Then

$$\alpha(t) = \frac{1}{\sum_{j=1}^{p} y_j^2(t)} \quad (19)$$

As can be seen, the learning rate is adapted to every pattern Y.

At the second phase it is necessary to compute $\Delta w_{ij}(t)$ in order to minimize the total mean-square error between the data Y and the target data set $\bar{Y}$ (equation 6). Then we have that

$$w_{ij}(t+1) = w_{ij}(t) - \alpha(t) \frac{\partial E'}{\partial w_{ij}(t)}. \quad (20)$$

The error function for a pattern

$$E' = \frac{1}{2} \sum_{j=1}^{n} (y_j - \bar{y}_j)^2 \quad (21)$$

where $\bar{y}_j$, $j = \overline{1, p}$, are the first principal components.

By using the gradient descent method we can get that

$$w_{ij}(t+1) = w_{ij}(t) - \alpha(t)(y_j - \bar{y}_j) x_i \quad (22)$$

The learning rate is

$$\alpha(t) = \frac{1}{\sum_{j=1}^{n} x_j^2(t)}. \quad (23)$$

5. Nonlinear networks

The nonlinear activation function is used for such network. By using nonlinearity of better results can be achieved in comparison with the linear function. Let's consider the training rules for nonlinear networks. As the activation function we will use a hyperbolic tangent. Then the j^{th} output of a forward layer is given by

$$y_j = th(s_j), \quad (24)$$

where $s_j = \sum_{i=1}^{n} w_{ij} x_i$, (25)

where $j = \overline{1, p}$.

Accordingly for an inverse layer

$$\bar{x}_i = th(s_i) \quad (26)$$

$$s_i = \sum_{j=1}^{p} w_{ji} y_j \quad (27)$$

Let's examine the training rules. At the first phase it is necessary for each input vector x to define such vector y, which will ensure the minimization of expression (4).

The error derivate is equal to

$$\gamma_j = \frac{\partial E}{\partial y_j} = \frac{\partial E}{\partial \bar{x}_i} \frac{\partial \bar{x}_i}{\partial s_i} \frac{\partial s_i}{\partial y_j} = \sum_i (\bar{x}_i - x_i)(1 - \bar{x}_i^2) w'_{ji} \quad (28)$$

Then

$$y_j(t+1) = y_j(t) - \alpha_j(t)\sum_i (\bar{x}_i - x_i)(1-\bar{x}_i^2)w'_{ji}. \quad (29)$$

By using Taylor series decomposition and the steepest descent method we can receive the adaptive training rate:

$$\alpha_j = \frac{\sum_{i=1}^{n}(\bar{x}_i - x_i)w'_{ji}}{\gamma_j \sum_{i=1}^{n}(w'_{ji})^2}, \quad (30)$$

Let's define the expression for the modification of the weights W'. Then:

$$\frac{\partial E}{\partial w'_{ji}} = \frac{\partial E}{\partial \bar{x}_i}\frac{\partial \bar{x}_i}{\partial S_i}\frac{\partial S_i}{\partial w'_{ji}} = (\bar{x}_i - x_i)(1-\bar{x}_i^2)y_j. \quad (31)$$

By using Taylor series decomposition and the steepest descent method we can receive the adaptive training rate:

$$\alpha(t) = \frac{\sum_{i=1}^{n}(\bar{x}_i - x_i)^2(1-\bar{x}_i^2)}{(\sum_{j=1}^{p} y_j^2)\sum_{i=1}^{n}(\bar{x}_i - x_i)^2(1-\bar{x}_i^2)^2}, \quad (32)$$

Then the modification of weight connections W' is as follows:

$$w'_{ji}(t+1) = w'_{ji}(t) - \alpha(t)(\bar{x}_i - x_i)(1-\bar{x}_i^2)y_j. \quad (33)$$

Thus at the first stage of the training the weights of the inverse layer w_{ji} and p - first principal components $\bar{y}_j$ are calculated.

At the second phase the weights of the forward layer are calculated, where the values $\bar{Y}$ are used as target outputs. For this purpose it is necessary to minimize the mean-square error (equation 6).
By using the gradient descent method we can get:

$$\frac{\partial E'}{\partial w_{ij}} = \frac{\partial E'}{\partial y_j}\frac{\partial y_j}{\partial S_j}\frac{\partial S_j}{\partial w'_{ij}} = (y_j - \bar{y}_j)(1-y_j^2)x_i, \quad (34)$$

By using Taylor series decomposition and the steepest descent method we can get the adaptive training rate:

$$\alpha(t) = \frac{\sum_{j=1}^{p}(y_j - \bar{y}_j)^2(1-y_j^2)}{(\sum_{i=1}^{n} x_i^2)\sum_{j=1}^{p}(y_j - \bar{y}_j)^2(1-y_j^2)^2}. \quad (35)$$

Then

$$w_{ij}(t+1) = w_{ij}(t) - \alpha(t)(y_j - \bar{y}_j)(1-y_j^2)x_i. \quad (36)$$

6. Proposed algorithm

Let L be the quantity of input patterns. The algorithm of sectioning training consist of the following steps:

1. Randomly initialization of weights. Choice of the minimum total mean-square error E_m.

2. Cosequently L of patterns enter the neural network inputs. By this for each pattern only a feed-forward transformation of data is performed. As a result of the given stage the set of the compressed images Y is defined which will be used at the next stage of the algorithm.

3. Only the inverse layer is considered. As the input information the values Y are used which are defined on the previous step of algorithm. As the target patterns of the inverse layer the vector of input data X is used. The following sequence of operations is performed:

3.1. For L patterns Y weights update using the expression (18) for linear or (33) for nonlinear neural networks.

3.2. For L patterns Y outputs update of the hidden layer using expression (16) for linear or (29) nonlinear neural networks.

3.3. The steps 3.1 and 3.2 are repeated, until the total mean-square error of an inverse layer becomes smaller than the given E_m.

4. The modification of the weight of the forward layer is performed according to expression (22) for linear or (36) nonlinear neural networks. For this purpose the input patterns consequently enter at the network and there is only forward transformation of the information for each pattern. The values $\bar{Y}$ obtained on the previous step of algorithm are used as the target data.

5. The step 4 proceeds untill the total mean-square error of the forward layer becomes smaller than the given one (E_m).

Figure 2. Fragments of the test image(at the top) and decompressed image (at the down).

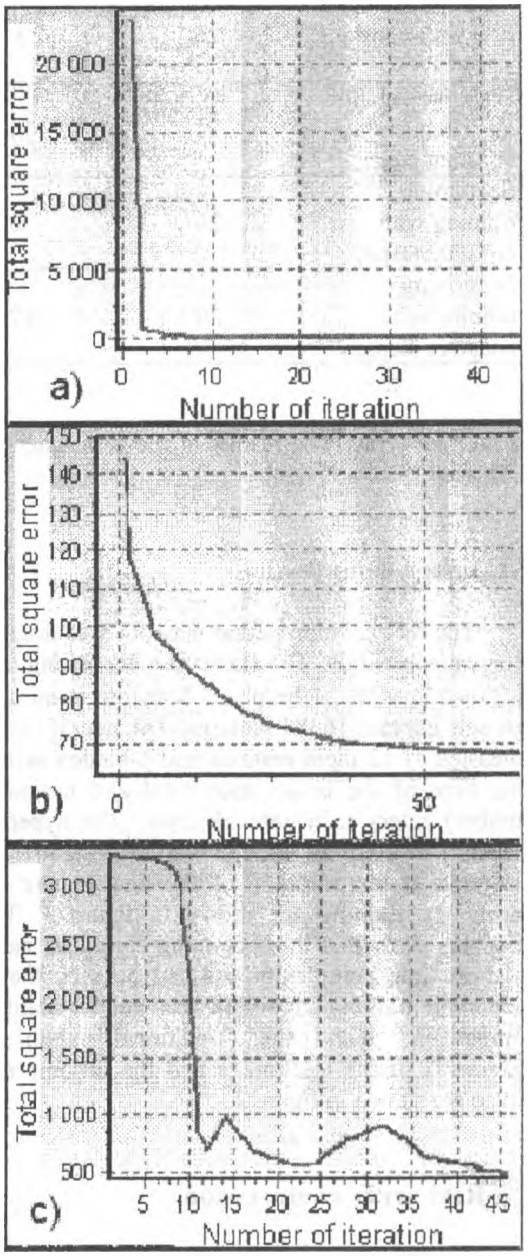


Figure 3. Diagrams of the variation of the total square error during training:
a) sectioning training; b) back-propagation algorithm; c) cumulative delta rule.

For definition of weights the normalized training rule can be used.

By using this algorithm it is possible to train the nonlinear recirculation network for the compression of the data. The experiments have shown that the offered algorithm is more effective in comparison with the usual ones [5-7].

7. Experiments

We have applied the proposed algorithm to the task of data compression. In the first experiment we illustrate the application of our results for the compression of human faces. In the second experiment our algorithm is used for the compression

Table 3. Comparison of different training methods.

Training method	Number of tests	Number of successful tests	Number of failure[1] tests	Average number of iterations	Average time of training, sec.
Cumultative delta rule with constant steps	20	3	17	2,07E4	27,4
Back propagation algorithms with constant steps	20	14	6	3,09E3	41,2
Sectioning training with constant steps	20	20	20	899	1,7
Sectioning training with adaptive steps	20	17	3	19,2	0,0412

of the real array. The results of the experiments are discussed.

7.1. Image compression

The recirculation neural network was tested for image compression. The standard color testing image is 256×256 with 24 bit/pixel. Both the training and test sets contain 16384 patterns. The neural network consisted of 12 input neurons and 6 hidden neurons. One byte of the information (real and normalized number) enters each network input. The hyperbolic tangent is used as the activation function. The diagrams of the variation of the total square error during the training are shown in figure 3. These diagrams show that the sectioning algorithm has the smallest time complexity. As can be seen that the sectioning training algorithm has good stability in comparison with the traditional algorithms. Fragments of the test image and the decompressed image are shown in figure 2.

7.2. Real array compression

For the analysis of the abilities of our training method we tested recirculation neural network for compression of the real arrays. In this tests we used different real arrays as the input data. The example of one of them is presented in table 1. Table 2 shows the compressed array after neural network training. In this case we use neural network with 4 input neural elements and 2 hidden ones. Also the hyperbolic tangent was used as the activation function. The total main square error for a successful test must be less than 0,01. Table 3 shows information about different training methods for compression of the given real array.

Table1. Input array.

0,134	0,045	0,032	0,032
0,135	0,134	0,032	0,023
0,072	0,073	0,144	0,032
0,025	0,125	0,123	0,123

Table 2. Compressed array for sectioning training with adaptive steps.

0,211	0,613
0,332	0,8

8. Conclusion

Our sectioning algorithm is an efficient tool for reducing of the dimension of the data. This algorithm will be used for training of the recirculation neural networks. We are currently exploring other applications of our approach.

9. Acknowledgments

This work is performed within the project INTAS 97-0606 "Development of an intelligent sensing instrumentation structure". The authors are thankful to the European Community for the financial support of the project.

10. References

1. Jolliffe I.T. Principal Component Analysis. // Springer-Verlag. 1986.
2. E. Oja, H. Ogawa, and J. Wangviwattana. Pca in fully parallel neural networks. In Aleksander

[1] if a total main square error reduced less than 1E-5 for 100 iteration.

&Taylor, editor, Artificial Neural Networks,2, 1992.
3. T.D. Sanger. Analysis of the two-dimensional receptive fields learned by the generalized hebbian algorithm in response to random input. Biological Cybernetics, 1990.
4. C. Fyfe and R. Baddeley. Non-linear data structure extraction using simple hebbian networks. Biological Cybernetics, 72(6):533{541, 1995.
5. Cottrell G., Munro P., Zipser D. Image compression by back-propagation: a demonstration of extensional programming // Tech. Rep. N.TR8702.–USCD: Institute of Cognitive Sciences –1987
6. Hinton G., McClelland J. Learning Representation by Recirculation // Proceedings of IEEE Conference on Neural Information Processing Systems.–1989.
7. Cottrell G., Munro P., Zipser D. Learning Internal Representation from Gray–Scale Images: An Example of Extensional Programming // Proceedings 9th Annual Conference of the Cognitive Science Society – 1987.–P.461–473.

Neural Network Model of Associative Memory: To Visualize Solutions in Weight Space

Akira Imada

Department of Electrical and Electronics Engineering
Anadolu University
Eskisehir, Turkey
E-mail: akira@mmf.mm.anadolu.edu.tr

Abstract

We apply some variants of evolutionary computations to the Hopfield model of associative memory. In the model, a number of patterns can be stored in the network as attractors if synaptic weights are determined appropriately. One of our goals of this study is to learn the number and distribution of these solutions in weight space, which is still an open problem. To address this issue, we test a method to visualize solutions in high-dimensional space in this paper.

keywords: Neural network, Associative memory, Weight space, Visualization.

1 Introduction

In studies using evolutionary algorithms, visualization of high-dimensional space provides various aspects of insight into the search space explored. We can imagine, for instance, convergence/divergence behaviors of a population, topology of a fitness landscape, what does a walk from a random point to the global optimum look like, and so on. The problem of mapping a number of points in multi-dimensional space to points in 2D space with the distances among the original points remaining as much as possible is one of those techniques. Shine et al. [1] and Collins [2] argued such a technique together with other possible alternatives. Collins call this technique *"Sammon Mapping"* after Sammon [3] who proposed this technique originally (Shine et al. call this *"Distance Map"*). Since the technique is an optimization problem, we can employ a genetic algorithm (GA) to solve this problem. Here we employ this technique in somewhat of a different way.

We apply some variants of evolutionary computations to the fully-connected neural network model of associative memory. In the model, a number of patterns can be stored in the network as attractors if synaptic weights are determined appropriately. Although some of the solutions of weights have been found heuristically, the number and distribution of the whole solutions are still unknown issue. As a preliminary stage toward addressing this issue, we apply the Sammon Mapping to visualize our wight space.

Since neither Collins nor Shine gave us any description such as how large dimensionality can be explored, or how many points can be mapped properly, we start by visualizing two known shapes in the space of high dimensionality. Then we apply the technique to our weight space of the neural network model of associative memory.

2 Associative Memory

Associative memory is a dynamical system which has a number of stable states with a domain of attraction around them [4]. If the system starts at any state in the domain, it will converge to the stable state. Hopfield [5] proposed a fully connected neural network model of associative memory in which information is stored by being distributed among neurons, and we can retrieve the information from dynamically relaxed neurons' states. In the model, some of the appropriate configurations of synaptic weights give the network a function of associative memory.

The Hopfield model consists of N neurons and N^2 synapses. Each neuron state is either active (+1) or

quiescent (-1). When an arbitrary N-bit *bipolar pattern*, a sequence of $+1$ and -1, is given to the network as an initial state, the dynamical behavior of neurons' states afterwards are characterized by the strengths of the N^2 synapses. The synaptic strengths are called *weights*, and the weight from neuron j to neuron i is denoted as w_{ij} in this paper. Provided the synaptic weights are determined appropriately, network can store some number of patterns as attractors. Hopfield employed the so-called Hebbian rule [6] to prescribe the weights. That is, to store p bipolar patterns ξ^ν

$$\xi^\nu = (\xi_1^\nu, \cdots, \xi_N^\nu), \quad \nu = 1, \cdots, p,$$

the weight values are determined as:

$$w_{ij} = \frac{1}{N} \sum_{\nu=1}^{p} \xi_i^\nu \xi_j^\nu \quad (i \neq j), \quad w_{ii} = 0.$$

An instantaneous state of a neuron is updated asynchronously (one neuron at a time) as:

$$s_i(t+1) = sgn\left(\sum_{j \neq i}^{N} w_{ij} s_j(t) \right), \qquad (1)$$

where $s_i(t)$ is a state of the i-th neuron at time t. If an initial state converges to one of the stored patterns ξ^ν as an equilibrium state, then the pattern is said to be *recalled*. Furthermore, if an initial state chosen from the stored patterns remains unchanged from the beginning, then the pattern is said to be stored as a *fixed point*.

In analyzing the Hopfield model, there have been basically two different approaches: one is to explore *pattern space* searching for attractors under a specific weight configuration, and the other is to explore *weight space* searching for an appropriate weight configuration that stores a given set of patterns. To be more specific, the former is an analysis of the Hamiltonian energy as a function of all the possible configurations of bipolar pattern to be given to the network, where synaptic weights are pre-specified using a learning algorithm, usually the Hebb's rule, so that the network stores a set of p given patterns. In this context, the model for $p = 1$ corresponds to the Mattis model of spin-glass [7], in which the Hamiltonian energy has two minima, while the model for infinitely large p corresponds to the Sherrington-Kirkpatrick model [8], in which the synaptic weights become Gaussian random variables. Analyses of the former type have been made in between these two extreme cases (see Amit [9]). The latter analysis, on the other hand, was addressed by Gardner [10]. She discussed the optimal weight configurations for a *fixed* number of given patterns in terms of the volume of the solutions in weight space, suggesting that the volume shrinks to vanish when p approaches to $2N$. In short, the former approach searches for the optimal pattern configurations which minimize the Hamiltonian energy *in pattern space with the weights being fixed*, while the latter searches for the weight configurations *in weight space* that optimally store a set of given *fixed patterns*.

In this paper, our concern is on weight space where some points give a network a capability to store a fixed set of patterns, and we call these points solutions.

3 Sammon mapping

As Collins [2] wrote, the dimension reduction has been an important technique for visualization of the space of high dimensionality. The Sammon Mapping [3] is one of these techniques. This enables us to map a set of $\mathcal{N}$ points in n-dimensional space to 2-D location data so that the distance information is preserved as much as possible, or as Shine [1] wrote *"so that the n-dimensional distances are approximated by 2 dimensional distances with a minimal error."* This problem is an optimization problem.

Shine et al. [1] and Collins [2] proposed a method to solve this problem by a Genetic Algorithm, as follows. First, the distance matrix whose entries are Euclidean distances between all possible pairs of $\mathcal{N}$ points in the n-dimensional space is calculated. Then tentative $\mathcal{N}$ points in 2-dimensional space are determined representing the original $\mathcal{N}$-points in the n-dimensional space. The distance matrix of these $\mathcal{N}$ 2D points is also calculated, which then will be subtracted from the original n-dimensional distance matrix, yielding an error matrix. A GA is used to minimize this error matrix.

For the sake of simplicity, we assume here the dimension reduction from 2401D space to 2D space. Given $\mathcal{N}$ points in 2401D space

$$X^1, \cdots, X^{\mathcal{N}},$$

where each point X^k is expressed by 2401 coordinates as

$$X^k = (x_1^k, \cdots, x_{2401}^k).$$

Then the square distance between m-th point and

n-th point is calculated as

$$R^{mn} = \sum_{i=1}^{2401} (z_i^m - z_i^n)^2$$

The values for all the possible combination of m and n construct a distance matrix. Since the matrix is symmetric with zero diagonal elements, we use the lower triangle elements ($m < n$).

Then we generate $\mathcal{P}$ sets of the 2-dimensional $\mathcal{N}$ points at random, i.e.,

$$\Psi^1(1), \cdots, \Psi^{\mathcal{N}}(1)$$
$$\Psi^1(2), \cdots, \Psi^{\mathcal{N}}(2)$$
$$\Psi^1(\mathcal{P}), \cdots, \Psi^{\mathcal{N}}(\mathcal{P})$$

where the k-th point of the i-th set is represented as:

$$\Psi^k(i) = (\zeta_1^k, \zeta_2^k).$$

Thus the i-th set of these $\mathcal{N}$ points has its distance matrix whose elements are

$$r^{mn}(i) = \sum_{i=1}^{2} (\zeta_i^m - \zeta_i^n)^2$$

The objective function of the i-th sets $f(i)$ can be defined as

$$f(i) = \sum_{m<n} r^{mn}(i) - R^{mn}$$

Starting with a random configuration of $\mathcal{N}$ points in 2-dimensional space, the GA corrects these points generation by generation applying crossover and mutation[1] to 2D coordinates. The correction is repeated until the error converges to an acceptable minimum.

4 Results & Discussion

Towards the goal of visualizing solutions in weight space, we apply the dimension-reduction technique to two toy examples, as test functions, in which distribution of the point is known. One is a set of points on a hyper-line, and the other is a set of points in the two separate regions. Our experiments of the fully-connected neural network model of associative memory are carried out on networks with 49 neurons, which implies the weight space is $49^2 = 2401$ dimensional space. So the dimensionality is set to 2401 in this paper.

[1] We employ uniform crossover [11] and BGA mutation [12] here.

Hyper-line

The first test is a visualization of a hyper-line. In

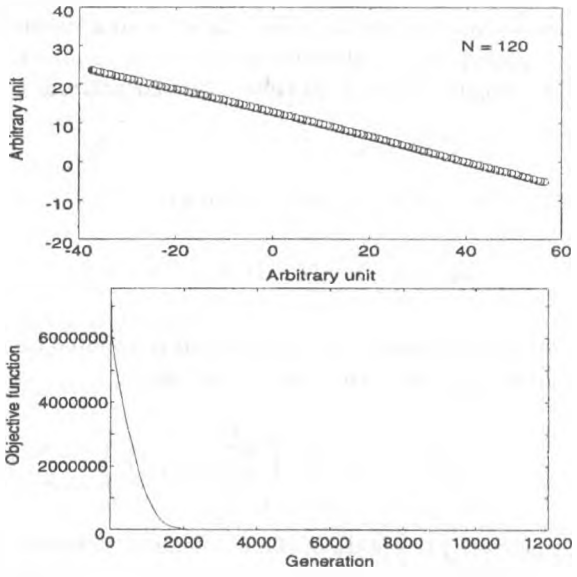


Figure 1: Points mapped to 2D space from 120 points on a diagonal line of the 2401-D space (top), and the time evolution of objective function (bottom).

mapping some points in high dimensionality to points in 2D space, there exists some constraints in general. It is clear, for example, that the four vertices of a tetrahedron in 3D space can never been exactly mapped to four points in 2D space. On the other hand, there is no such constraint in the case of points on a hyper-line. In that sence, hyper-line is a good benchmark for the algorithm.

First, we pick up 120 points that are distributed with equal interval on a diagonal line of the 2401-dimensional hyper-cube. To be more specific, the points are:

$$(x_1^k, x_2^k, \cdots, x_{2401}^k), \quad k = 0, 1, \cdots, 119$$

where

$$x_1^k = x_2^k = \cdots = x_{2401}^k = -1 + k \cdot (2/119)$$

Then they are mapped to 120 points on 2-dimensional space so that the distance relation among the 120

points on the 2401-dimensional space is kept as much as possible. A result is shown in Figure 1 (top). We can see a straight line in the 2-dimensional space. The task to search for an appropriate configuration of 2D points is quite easy in this case. As evolution proceeds, the objective function that took the value 7,560,777 at the start asymptotically approaches the small value around 0.1. The evolution is shown in Figure 1 (bottom).

Two hyper-cubes

Next, we proceed to an example in which we can imagine the shape of the region in high-dimensional space. We sampled 60 points randomly from the 2401 dimensional region whose ordinates are all between 0.5 and 1.0 as well as the other 60 points from the region whose ordinates are between -0.5 and -1.0. Namely, points are sampled either from two separate hyper-cubes of the same size. In Figure 2 (top), a result of dimension reduction of these 120 points is shown, together with a point that corresponds to the origin. The evolution of the objective function is shown in Figure 2 (bottom). The value starts with 14,567,428, and eventually approaches 106,125. Though the final value of the objective function is not so small, the ratio of the final value is 0.7% of the initial value. We can clearly see the two separate regions in the 2-dimensional space.

Hyper-sphere: weight solution

As stated earlier, multiple configurations of weights give a network a function of associative memory. The number of these configurations is known to be dependent on p, the number of patterns to be stored. Storing just one pattern gives a maximum number of solutions of weights, while as p approaches twice the number of neurons, all the solutions vanishes [10]. However, the number and distribution as a function of p is still unknown. So far, we have applied some variants of evolutionary algorithms to search for these solutions (see e.g., [13]– [18]). Here, we study the solutions found by the Breeder Genetic Algorithm (BGA) among others, since only this algorithm has been able to search for solutions for a wide range of p (see [19]). Our experiments were carried out on networks with 49 neurons and the BGA found solutions for up to $p = 90$. The solutions that the BGA found are also expected to be different from run to run, as Mühlenbein et al. [12] wrote: *"the BGA mutation scheme is able to optimize many multi-modal functions."* As a preliminary stage of the goal of learning the number and distribution as a function of p, we

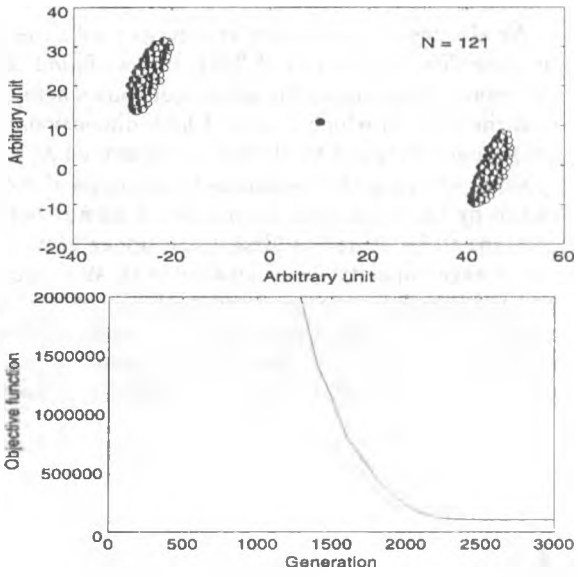


Figure 2: Two regions of the 2401-D space mapped to the 2-D space; filled-in circle • indicates the origin (top), and the time evolution of objective function (bottom).

sampled 30 such solutions for $p = 1$. It is important to note here that since each weights, w_{ij}, can take an arbitrary real value, there are infinite number of equivalent configurations which differ only by scaling factor. In other words, for any scaling factor κ, κw_{ij} works exactly in the same way as w_{ij} in updating neuron states (see equation (1)). So, we normalized the solutions obtained such that they locate on the hyper-sphere of radius 1. We suspect that these normalized solutions for $p = 1$ are distributed uniformly on the surface of the hyper-sphere. We show the results of the 2D points mapped from the 2401-dimensional solution space for the number of solutions $\mathcal{N} = 9$ in Figure 3 (top). When the number of storing patterns is only one, we observed that the nine 2D points corresponding to the solutions are almost uniformly distributed on the circle whose center corresponds to the origin of 2401-dimensional space, while the distribution of these 2D points are disturbed more or less for $\mathcal{N}$ more than 9 (not shown here). In Figure 4, we show the time evolution of each objective function value for $\mathcal{N} = 9$ and 30. The value for $\mathcal{N} = 9$ starts with 1,758 while the value for $\mathcal{N} = 30$ starts with 37,115 and these values ended up 8 and 129, respectively. The difference of these values is due to the degree of constraint of the dimension reduction

problem.

We also tested the similar experiment with the dimensionality 256 instead of 2401, but we found that the results were almost the same (not shown here), in that the limit in which points of high dimensionality are properly mapped to 2D space was around $\mathcal{N} = 9$.

Next, we apply the technique to solutions also obtained by the BGA runs for $p = 90$. This number of patterns to be stored is almost the upper bound of the storage capacity for a network with 49 neurons, and the solutions are expected to be localized into small region of weight space [10]. A result is shown in Figure 3 (bottom). Though resolution is not so good, we can anyhow imagine the localized solutions.

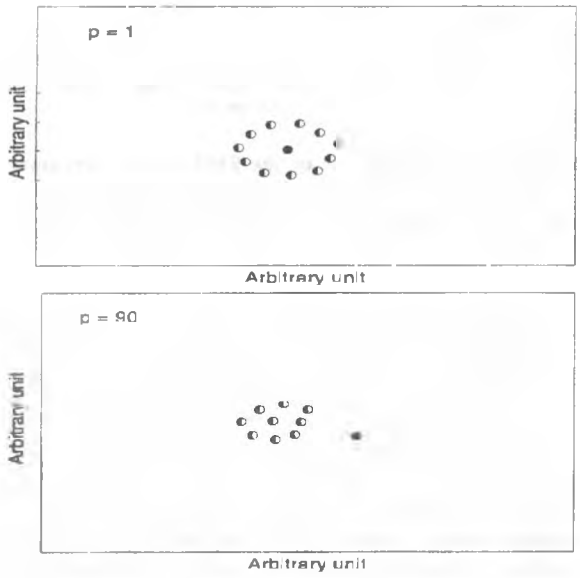


Figure 3: 2D points mapped from nine solutions in the 2401-dimensional weight space. Filled-in circle • indicates the origin. The number of stored patterns is 1 (top) and 90 (bottom).

5 Conclusion

We have described a technique to map a high-dimensional search space to 2D points remaining the distance information of the source points being as much as possible. What we are interested in is to visualize weight configurations in weight space that give a network a function of associative memory. Since the topic has not been the subject of extensive research, as Shine et al. [1] pointed out, we preliminary applied the technique to two somewhat of trivial examples to know the limitation of the technique. We have observed that 120 points on a 2401-dimensional hyper line are mapped to a straight line on 2D space, and 120 points that are distributed randomly on two separate hyper-cubes in 2401-dimensional space are mapped to two separate regions of 2D points.

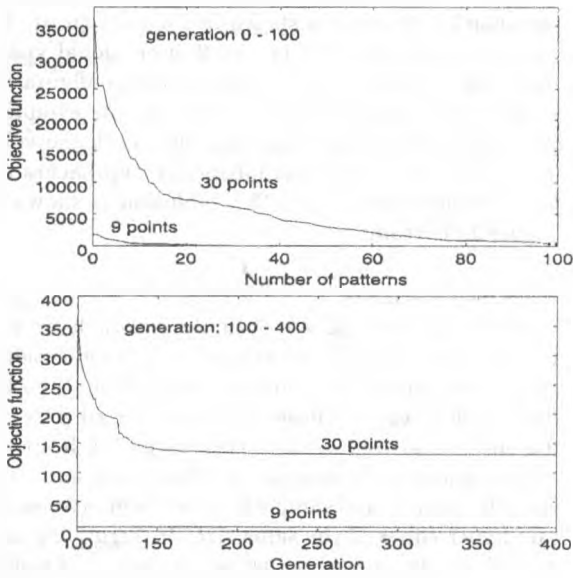


Figure 4: Time evolution of the objective function when high dimensional points are distributed on a hyper-sphere.

Then we apply the technique to our weight space in which we search for the solutions that give a network a function of associative memory. The dimension of the weight space is 2401 since we use neural networks with 49 neurons. If p, the number of patterns to be stored to the network as an associative memory, is only one, then the solutions are expected to be uniformly distributed in the space. On the other hand, if $p = 90$ that is almost the upper bound of the storage capacity, then the solutions are expected to be highly localized. We normalized these solutions such that they locate on the hyper-sphere of radius one. Then we used the technique to visualize these solutions in 2D space, and found that we can obtain the expected results unless the number of solutions exceeds nine. This limit is not so good, but the result suggests that the solutions are distributed uniformly at random for $p = 1$ and very localized for $p = 90$.

References

[1] W. B. Shine, and C. F. Eick (1997) *Visualizing the Evolution of Genetic Algorithm Search Processes.* Proceedings of the IEEE International Conference on Evolutionary Computation, pp.367–372.

[2] T. D. Collins (1997) *Using Software Visualization Technology to Help Evolutionary Algorithm Users Validate their Solutions.* Proceedings of the 7th International Conference on Genetic Algorithms, pp.307–314.

[3] J. W. Sammon (1969) *A Nonlinear Mapping for Data Structure Analysis.* IEEE Transactions on Computers. C-18(5), pp.401–408.

[4] J. Komlós, and R. Paturi (1988) *Convergence Results in an Associative Memory Model.* Neural Networks, 1, pp.239–250.

[5] J. J. Hopfield (1982) *Neural Networks and Physical Systems with Emergent Collective Computational Abilities.* Proceedings of the National Academy of Sciences, USA, 79, pp.2554-2558.

[6] D. O. Hebb (1949) *The Organization of Behavior. Wiley.*

[7] D. C. Mattis (1976) *Solvable Spin Systems with Random Interactions.* Physics Letters, 56A, pp.421–422.

[8] D. Sherrington, and S.Kirkpatrick (1975) *Solvable Model of a Spin Glass.* Physical Review Letters, 35, pp.1792–1796.

[9] D. J. Amit (1989) *Modeling Brain Function: The World of Attractor Neural Networks.* Cambridge University Press.

[10] E. Gardner (1988) *The Phase Space of Interactions in Neural Network Models.* Journal of Physics, 21A, pp.257–270.

[11] G. Syswerda (1989) *Uniform Crossover in Genetic Algorithms.* Proceedings of the 3rd International Conference on Genetic Algorithms, pp.2–9.

[12] H. Mühlenbein, and D. Schlierkamp-Voosen (1996) *Predictive Models for the Breeder Genetic Algorithm I. Continuous Parameter Optimization.* Evolutionary Computation, 1, pp.25–49.

[13] A. Imada, and K. Araki (1995) *Genetic Algorithm Enlarges the Capacity of Associative Memory.* Proceedings of the 6th International Conference on Genetic Algorithms, pp.413–420.

[14] A. Imada, and K. Araki (1996) *Lamarckian Evolution of Associative Memory.* Proceedings of IEEE International Conference on Evolutionary Computation, pp.676–680.

[15] A. Imada, and K. Araki (1997) *The Baldwin Effect on the Evolution of Associative Memory.* Proceedings of the 3rd International Conference on Artificial Neural Networks and Genetic Algorithms, pp.354–358.

[16] A. Imada, and K. Araki (1997), *Searching Real-Valued Synaptic Weights of Hopfield's Associative Memory using Evolutionary Programming.* Proceedings of the 6th Annual Conference on Evolutionary Programming, Springer-Verlag, Lecture Notes in Computer Science, No.1213, pp.13–22.

[17] A. Imada, and K. Araki (1997), *Application of an Evolution Strategy to the Hopfield Model of Associative Memory.* Proceedings of the IEEE International Conference on Evolutionary Computation, pp.679–683.

[18] A. Imada, and K. Araki (1997), *Random Perturbations to Hebbian Synapses of Associative Memory using a Genetic Algorithm.* Proceedings of International Work-Conference on Artificial and Natural Neural Networks, Springer-Verlag, Lecture Notes in Computer Science No.1240, pp.398–407.

[19] A. Imada, and K. Araki (1997) *Evolution of Hopfield Model of Associative Memory by the Breeder Genetic Algorithm.* Proceedings of the 7th International Conference on Genetic Algorithms, pp.784–791.

New Approach of the Recurrent Neural Network Training

V. Golovko, Y. Savitsky
Brest Polytechnic Institute
Moskowskaja str. 267, Brest 224017
Belarus, e-mail: cm@brpi.belpak.brest.by

Abstract

In this work the technique of creation of adaptive training algorithms for recurrent neural networks (RNN) is considered. These algorithms have high convergence and accuracy on a comparison with traditional backpropagation. The original technique of calculation of an adaptive training step with use of steepest descent method is resulted. The features of calculation of an adaptive pitch for neural elements with recurrent connections are discussed. Are considered the neural units with various functions of activation, used in architectures neural systems of forecasting. The indicated computing experiments demonstrate advantage of the developed RNN training methods.

1: Introduction

The training of RNN with plenty of recurrent connections with use backpropagation is a large problem. It is connected to defects of training algorithm and complexity of neural network architecture. Therefore there is a problem of development of adaptive training algorithms permitting to execute RNN training of a complicated structure with a high exactitude and a speed [3-6]. In this work the technique of calculation of an adaptive training step is resulted which can be applied to any activation of neural elements. The adaptive step for sigmoid, logarithmic and linear functions of activation used by the authors for construction of systems of forecasting is considered.

2: Basic RNN Architecture

This work is focused in the fully connected third-layered recurrent neural network (RNN) architecture, as shown in a Fig 1 [3]. The output activity of the neural network is defined by expression:

$$Y_j(t) = F\left(\sum_{i=1}^{N_h} w_{ij} y_i(t) - T_j \right) \quad (1)$$

where N_h - number of units of the hidden level, $y_i(t)$ - output activity of hidden units, s_j - threshold for output unit, w_{ij} - weights from hidden input units i to the output unit j, $j = \{1, N_o\}$.

The output activity of the hidden units on the current training iteration t for training exemplar p is defined as:

$$y_j(t) = F\left(\sum_{i=1}^{N_I} w_{ij} x_i(t) + \sum_{k=1}^{N_h} w_{kj} y_k(t-1) + \sum_{l=1}^{N_o} w_{lj} Y_l(t-1) - s_j \right) \quad (2)$$

where $x_i(t)$ is a i'th element of the input vector $x(t)$, N_I - size of an input vector, w_{ij} - weights from external units i to hidden units j, w_{kj} - weights from hidden units k to hidden units j, $y_k(t-1)$ - output activity of a hidden unit k for the previous moment of time, w_{lj} - weight to the hidden unit j from an output unit l, $Y_l(t-1)$ - network output activity for the previous moment of time and s_j - thresholds of the hidden units.

In this work we use three types of neural units transfer function:

- *Linear* activation function:

$$F(S_j) = M \cdot S_j \quad (3)$$

This function is used as RNN output units in the some types of the neural prediction systems.

- *Logarithmic* activation function:

$$F(S_j) = \ln\left((S_j + \sqrt{(S_j)^2 + a}) / \sqrt{a} \right), \ (a > 0) \quad (4)$$

The choice by this transfer function is stipulated by that it is unlimited on all define area. It allows better to simulate and predict complex non-stationary processes. We propose to use this function in the hidden units of RNN. The parameter a defines declination of the activation function (see Fig. 2).

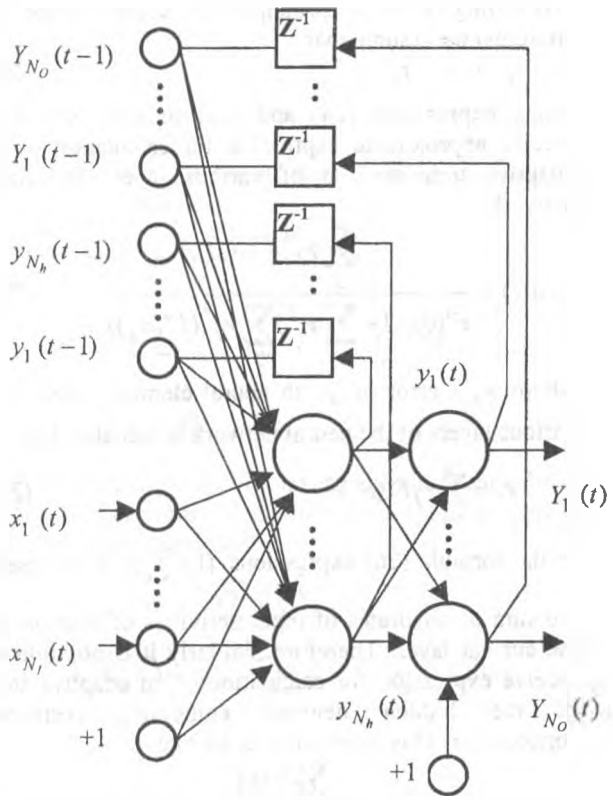


Figure 1. The RNN architecture

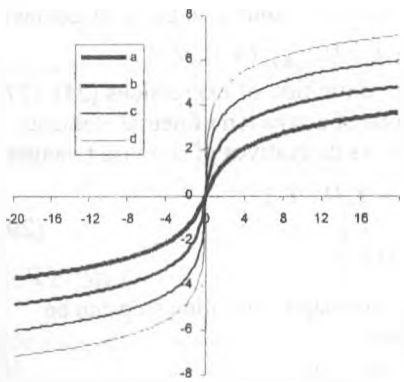


Figure 2. The logarithmic activation function for various parameters a: a) $a = 1.0$, b) $a = 0.1$, c) $a = 0.01$, d) $a = 0.001$

- At other case in hidden units is used standard *sigmoid* transfer function, defined as:

$$F(S_j) = \left((1 + e^{-S_j}\right)^{-1} \tag{5}$$

Let's consider the adaptive training step calculation for mentioned above transfer functions.

3: Adaptive Training Step Calculation for RNN

In standard backpropagation algorithm there is a problem of choice of an optimal training step to increase speed of training. For choice of adaptive step it would be possible to use a method of steepest descent. According to it, on each iteration of neural network training, the training step for each layer by such is necessary to select to minimize a root-mean-square error of a neural network:

$$\alpha(t) = \min E(y_j(t+1)) \tag{6}$$

where $j = \{1, N_O\}$, N_O - number of neural units of the output layer.

The output value of the j'th neuron depends on function of activation of units and is generally determined as follows:

$$y_j(t+1) = F(w_{ij}(t+1), T_j(t+1)) \tag{7}$$

The weights and thresholds are modified as:

$$w_{ij}(t+1) = w_{ij}(t) - \alpha(t)\frac{\partial E}{\partial w_{ij}(t)}, \qquad T_j(t+1) = T_j(t) - \alpha\ (t)\frac{\partial E}{\partial s_j(t)}, \tag{8}$$

Root-mean-square error of the RNN is defined as:

$$E = \frac{1}{2}\sum_{j=1}^{N_O}(Y_j - t_j)^2 \tag{9}$$

Then for determination $\alpha(t)$ it is necessary to find

$$\frac{\partial E}{\partial \alpha(t)} = \frac{\partial E}{\partial Y_j(t+1)}\frac{\partial Y_j(t+1)}{\partial \alpha(t)}. \tag{10}$$

The given equation cannot be decided rather $\alpha(t)$ by an analytical way. Therefore in the series of publications for definition of an adaptive training step it is offered to use methods of linear search [1]. However it is connected to difficult calculations. Therefore it is possible to offer an approximate method of a determination of a training step $\alpha(t)$. It bases on expansion of the neural unit activation function in a number Tailor serial. Let's consider it explicitly.

Let output value of the j'th neuron of the output layer is equaled:

$$Y_j(t) = F(S_j(t)), \qquad S_j(t) = \sum_i y_i(t) w_{ij}(t) - T_j(t), \tag{11}$$

where $y_i(t)$ - output value of the i'th neuron of the hidden layer.

For definition of the weighted sum of the j'th neuron in the moment of time $t+1$ we shall use in (11) expressions (8):

$$S_j(t+1) = \sum_i y_i(w_{ij} - \alpha\frac{\partial E}{\partial w_{ij}}) - T_j + \alpha\frac{\partial E}{\partial T_j} =$$
$$= \sum_i y_i w_{ij} - T_j - \alpha\cdot(\sum_i y_i\cdot\frac{\partial E}{\partial w_{ij}} - \frac{\partial E}{\partial T_j}). \quad (12)$$

We shall designate

$$a_j = \sum_i y_i\frac{\partial E}{\partial w_{ij}} - \frac{\partial E}{\partial T_j}. \quad (13)$$

Then the expression (12) can be presented as follows:

$$S_j(t+1) = S_j(t) - \alpha\cdot a_j. \quad (14)$$

The target value of j th neuron in the moment of time $t+1$ is equaled:

$$Y_j(t+1) = F(S_j(t+1)). \quad (15)$$

Let's decompose this expression under the Tailor formula and limits by first two members:

$$Y_j(t+1) = F(0) + F'(0)\cdot S_j(t+1), \quad (16)$$

where

$$F'(0) = \frac{\partial F}{\partial S_j} \text{ for } S_j = 0.$$

We shall use in (16) expressions (14). Then

$$Y_j(t+1) = F(0) + F'(0)S_j(t) - \alpha F'(0)a_j \quad (17)$$

As

$$Y_j(t) = F(0) + F'(0)S_j(t), \quad (18)$$

that expression (17) can be presented as follows:

$$Y_j(t+1) = Y_j(t) - \alpha F'(0)a_j \quad (19)$$

For definition of an adaptive training step it is necessary to supply:

$$E = \frac{1}{2}\sum_j (Y_j(t+1) - t_j)^2 \to \min \quad (20)$$

Then

$$\frac{\partial E}{\partial \alpha} = \sum_j (Y_j(t) - t_j - \alpha F'(0)a_j)\cdot(-F'(0)a_j) = 0 \quad (21)$$

Expressing from the last equation, we shall receive:

$$\alpha(t) = \frac{\sum_j (Y_j(t) - t_j)a_j}{F'(0)\sum_j a_j^2} \quad (22)$$

As $\frac{\partial^2 E}{\partial \alpha^2} > 0$, for want of given α the minimum of a root-mean-square error is ensured. Let's find expression for a_j. For this purpose we shall define:

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial Y_j}\cdot\frac{\partial Y_j}{\partial s_j}\cdot\frac{\partial S_j}{\partial w_{ij}} = (Y_j - t_j)F'(S_j)\cdot y_i,$$
$$\frac{\partial E}{\partial T_j} = \frac{\partial E}{\partial Y_j}\cdot\frac{\partial Y_j}{\partial S_j}\cdot\frac{\partial S_j}{\partial T_j} = -(Y_j - t_j)F'(S_j). \quad (23)$$

Using (22) in expression (12), we shall receive:

$$a_j = (1+\sum_i y_i^2)\cdot(Y_j - t_j)\cdot F'(S_j) \quad (24)$$

Proceeding from a principle of independence of stratums, we assume that

$$\gamma_j = Y_j - t_j \quad (25)$$

Using expressions (24) and (25) in (22), we shall receive approximate expression for calculation of an adaptive training step of various layers if neural network:

$$\alpha(t) = \frac{\sum_j \gamma_j^2 F'(S_j)}{F'(0)\cdot(1+\sum_j y_i^2)\sum_j \gamma_j^2 (F'(S_j))^2} \quad (26)$$

where γ_j - error of j 'th neural element, which for various layers of the neural network is calculated as:

$$\gamma_i = \sum_{j=1}^{N_{i+1}} \gamma_j F'(S_j)w_{ij} \quad (27)$$

In the formula (26) expressions $(1+\sum_i y_i^2)$ represent the sum of quadrates of input activities of neurons of the current layer. Therefore similarly it is possible to receive expression for calculation of an adaptive step of the hidden neurons containing recurrent connections. This expression looks like:

$$\alpha(t) = \frac{\sum_j \gamma_j^2 F'(S_j)}{F'(0)\cdot(1+\sum_i x_i^2(t) + \sum_k (y_k(t-1))^2 + \sum_l (y_l(t-1))^2)\sum_j \gamma_j^2 (F'(S_j))^2}, \quad (28)$$

where $y_k(t-1), y_l(t-1)$, - source activity of context-sensitive neurons, $k = \{1, N_h\}, l = \{1, N_O\}$.

Let's consider definition of expressions (25), (27) for various functions of activation of neural elements.

Sigmoid function. As derivatives of sigmoid function:

$$Y_j' = F'(S_j) = Y_j(1 - Y_j),$$
$$Y_j'(0) = F'(0) = \frac{1}{4}, \quad (29)$$

that expression for an adaptive training step can be presented as follows:

$$\alpha(t) = \frac{4\sum_j \gamma_j^2 Y_j(1 - y_j),}{(1+\sum_i y_i^2)\cdot\sum_j \gamma_j^2 Y_j^2 (1 - Y_j)^2} \quad (30)$$

For neurons of the hidden layer, with allowance for of recurrent connections, the adaptive training step is determined by expression:

$$\alpha(t) = \frac{4\sum_j \gamma_j^2 y_j(1-y_j),}{(1+\sum_i x_i^2 + \sum_k (y_k(t-1))^2 + \sum_l (y_l(t-1))^2)\cdot\sum_j \gamma_j^2 y_j^2 (1-y_j)^2} \quad (31)$$

Logarithmic function. Derivatives of logarithmic function is:

$$Y_j' = F'(s_j) = \frac{1}{\sqrt{S_j^2 + a}}.$$
$$Y_i'(0) = F'(0) = \frac{1}{\sqrt{a}}, \quad (32)$$

that expression for an adaptive training step can be presented as follows:

$$\alpha(t) = \frac{\sqrt{a}\sum_j \gamma_j^2 \left(\sqrt{(S_j)^2 + a}\right)^{-1}}{(1 + \sum_i y_i^2) \cdot \sum_j \gamma_j^2 \left((S_j)^2 + a\right)^{-1}}, \quad (33)$$

$$\gamma_i = \sum_j \gamma_j y_j (1 - y_j) w_{ij} \quad (34)$$

For neurons of the hidden layer, with allowance for of recurrent connections, the adaptive training step is determined by expression:

$$\alpha(t) = \frac{\sqrt{a}\sum_j \gamma_j^2 \left(\sqrt{(S_j)^2 + a}\right)^{-1}}{(1 + \sum_i y_i^2 + \sum_k (y_k(t-1))^2 + \sum_l (y_l(t-1))^2) \cdot \sum_j \gamma_j^2 \left((S_j)^2 + a\right)^{-1}}, \quad (35)$$

$$\gamma_i = \sum_j \gamma_j \frac{1}{\sqrt{S_j^2 + a}} w_{ij} \quad (36)$$

Linear function. As derivatives of linear function:

$$Y_j' = F'(S_j) = M,$$
$$Y_j'(0) = F'(0) = M, \quad (37)$$

then we receive the next expression:

$$\alpha(t) = \frac{M\sum_j \gamma_j^2}{M(1 + \sum_i y_i^2) \cdot \sum_j \gamma_j^2 M^2} = \frac{1}{M^2(1 + \sum_i y_i^2)} \quad (38)$$

4: Testing

For simulation two types of RNN architectures were taken. One of them consists from neurons with sigmoid activation function in the hidden layer. In the other case were used hidden units with logarithmic transfer function. Both neural networks contained 20 inputs, 5 nonlinear elements in the hidden layer and one output linear neuron. For training set organization were used the time serial of passenger airtransportations described in [2]. It size is 144 units. The testing task is forecasting of the described below time serial. For training both neural networks 3000 training iterations were executed. In procedures of training the appropriate expressions for calculation of an adaptive pitch were used. For a comparison the training with various constant steps were executed. The quality of training was inspected by a root-mean-square training error, as (9). In all cases higher efficiency of adaptive algorithms was observed (see Table 1).

Table 1. RNN training results

Type of training step	Training error for RNN with logarithmic transfer function a=0.01	Training error for RNN with sigmoid transfer function
Adaptive	$1.78E^{-5}$	$6.92E^{-4}$
$\alpha = 0.0099$	$3.91E^{-2}$	$3.11E^{-1}$
$\alpha = 0.099$	$7.13E^{-3}$	$5.01E^{-2}$
$\alpha = 0.99$	$9.11E^{-4}$	$9.78E^{-3}$
$\alpha = 1.99$	$5.21E^{-4}$	$6.97E^{-3}$
$\alpha = 9.99$	$2.87E^{-1}$	$9.98E^{-4}$
$\alpha = 19.99$	1.09	$8.78E^{-3}$

5: Conclusion

In this work the problem of fast training of recurrent multilayer neural networks with complicated structure is partially decided. The circumscribed original technique of calculation of an adaptive training step can be applied for any functions of activation of neuroelements. In work the application of the developed algorithms on an example of concrete types of RNN architectures is shown. The computing experiments demonstrate potential abilities of the developed adaptive algorithms for training of complicated neural networks.

Acknowledgments
This paper is supported by INTAS program "INTAS OPEN 97-0606". The authors express gratitude to the European Union for financial support.

6: References

1. Hortz J., Krogh A., Palmer R. Introduction to the theory of neural computation. – Addison Wesley Publishing Company.–1991.–327p.
2. Box G.E.P., Genkins G. M. Time-Series Analysis, Forecasting and Control. New York, 1970.
3. Pedersen M. Training Recurrent Networks // Proceeding of the IEEE Workshop on Neural Networks for Signal Processing VII. –New Jersey: IEEE.–1997.
4. T.Lin, B.G. Horne, P.Tino., C.L. Giles. Learning Loln-Term Dependies with NARX Recurrent Neural Networks. // IEEE Transactions on Neyral Networks., vol. 7, no. 6, p. 1329, - 1996.
5. M.W. Pedersen, L.K. Hansen. Recurrent Neural Networks: Second-Order Properties and Pruning. // Advanced in Neural Information Processing Systems 7, Cambridge, MA: The MIT Press, 1995, pp. 673-680.
6. Pineda F. Generalization of back-propagation to recurrent neural network. // Physical Review Letters, 19(59), 2229-2232.- 1987.

The Training of Feed-Forward Neural Networks

Golovko V., Dunets A., Savitsky Y.

Brest Polytechnical Institute, Mosckovskaja str., 267
224017, Brest, Republic of Belarus
E-mail: cm@brpi.belpak.brest.by

Abstract

The training of multilayer perceptron is generally a difficult task. Excessive training times and lack of convergence to an acceptable solution are frequently reported. This paper discribes new training methods of feedforward neural networks. In comparison with standard backpropagation algorithm it has smaller time complexity and better convergence. The testing of the proposed algorithm was carried out on a coding information task. The results of experiments are discussed.

Key words: neural networks, backpropagation, training, simulation.

1. Introduction

The basis of multilayer perceptron is backpropagation algorithm. It is characterised by unadaptive rate and instability of training. The instability of training depends on the initial initialisation of weights.

Many researches have devised improvements of and extensions to the basic backpropogation algorithm [1,2]. In paper [3] the backpropogation algorithm with training adaptive rate was offered. As a result the temporary complexity was reduced. However it has not solved the problem of stability of the training algorithm.

This paper describes a new approach for training of the feedforward neural networks. The results of experiments are discussed.

2. Theoretical bases of the method

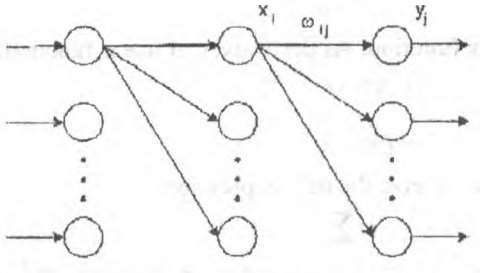


Fig 1. The general structure of the neural network

Hyperbolic tangent is used is this multilayer neural network (fig. 1) as the activation function of neural elements

$$y_j = th(S_j) = \frac{e^{S_j} - e^{-S_j}}{e^{S_j} + e^{-S_j}}, \qquad (1)$$

where S_j characterises the weighed sum j-th of the neural element. It is defined as follows:

$$S_j = \sum x_i \omega_{ij} - T_j, \qquad (2)$$

where T_j - the threshold j-th of the neural element, x_i - the output of the neural element of the previous layer, w_{ij} - the weight between neural elements i and j.

The root-mean-square error of the neural network for one pattern is defined as

$$E=\frac{1}{2}\sum_j (y_j-t_j)^2, \qquad (3)$$

where t_j - the desired target value for neurone j.

The standard method of backpropagation consists in use of gradient descent algorithm in space of weights and thresholds of the neural network:

$$\omega_{ij}(t+1)=\omega_{ij}(t)-\alpha\frac{\partial E}{\partial \omega_{ij}(t)}, \qquad (4)$$

$$T_j(t+1)=T_j(t)-\alpha\frac{\partial E}{\partial T_j(t)}, \qquad (5)$$

where α - the peed of training.

With the use of the steepest descent method it is possible to receive expression for an adaptive step of training of the neural network:

$$\alpha(t)=\frac{\sum_j \gamma_j^2(1-y_j^2)}{(1+\sum_i x_i^2)\sum_j \gamma_j^2(1-y_j^2)^2}, \qquad (6)$$

where γ_j - error j-th neural element. For the output layer

$$\gamma_j=y_j-t_j, \qquad (7)$$

and for other layers it is defined as:

$$\gamma_j=\sum_i \gamma_i(1-y_i^2)\omega_{ji}, \qquad (8)$$

Here i – the number of units of the following layer in relation to the layer j. According to it

$$\omega_{ij}(t+1)=\omega_{ij}(t)-\alpha(t)\gamma_j y_i(1-y_j^2), \qquad (9)$$

$$T_j(t+1)=T_j(t)+\alpha(t)\gamma_j(1-y_j^2), \qquad (10)$$

As it was already marked, the expression (6) allows considerably increasing speed of training. However it does not solved the problem of stability of the algorithm. For the neutralisation of this lack it is offered alongside with the modification of weights and thresholds of neural elements also to carry out the modification of outputs of hidden layer's neurones. So for hidden layer the outputs of the neural elements will change as follows:

$$x_i(t+1)=x_i(t)-\alpha_i(t)\frac{\partial E}{\partial x_i(t)}. \qquad (11)$$

Let's find the derivative of the error function on x_i

$$\frac{\partial E}{\partial x_i}=\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial S_j}\frac{\partial S_j}{\partial x_i}= \qquad (12)$$
$$=\sum_j (y_j-t_j)(1-y_j^2)\omega_{ij}=\gamma_i$$

For calculating of the training adaptive step $\alpha_i(t)$ we shall use the steepest descent method. Then

$$\alpha_i(t)=\min\left\{E\left[x_i-\alpha_i(t)\frac{\partial E}{\partial x_i(t)}\right]\right\}. \qquad (13)$$

Let's define the weighed sum as

$$S_j'=\omega_{ij}(x_i(t)-\alpha_i\gamma_i)+\sum_{k\neq i}\omega_{kj}x_k(t)-T_j, \qquad (14)$$

After the transformation of this expression we get

$$S_j'=S_j-\alpha_i\gamma_i\omega_{ij}. \qquad (15)$$

Using Taylor series decomposition for function $\gamma_j'=th(S_j')$ and transforming the received expression, we have

$$y_j'=y_j-\alpha_i\gamma_i\omega_{ij}. \qquad (16)$$

From here the root-mean-square error of the network

$$E=\frac{1}{2}\sum_j (y_j'-t_j)^2. \qquad (17)$$

We find such α_i, for which the root-mean-square error of the network is minimal

$$\frac{\partial E}{\partial \alpha_i}=\sum_j (y_j-\alpha_i\gamma_i\omega_{ij}-t_j)(-\gamma_i\omega_{ij})=0. \qquad (18)$$

Transforming last expression, we find the meaning of the training adaptive step the neurones of the hidden layer

$$\alpha_i=\frac{\sum_j (y_j-t_j)\omega_{ij}}{\sum_j \omega_{ij}^2\sum_j (y_j-t_j)(1-y_j^2)\omega_{ij}}. \qquad (19)$$

Let $y_j-t_j=\gamma_j$. Then

$$\alpha_i=\frac{\sum_j \gamma_j\omega_{ij}}{\gamma_i\sum_j \omega_{ij}^2}, \qquad (20)$$

where γ_j, γ_i - accordingly the error of j-th neurone of the following layer and the error of i-th neurone of the previous layer.

Thus it is supposed to carry out the training of the neural network on three parameters: weights, thresholds and outputs of neural elements. The outputs of the neural elements change with the purpose of minimisation of the neural network root-mean-square error as follows

$$x_i(t+1) = x_i(t) - \alpha_i(t)\gamma_i, \qquad (21)$$

where γ_i - error i-th of the neural element.

As the area of meanings of the activation function of neural elements is segment D = [-1;1], after the reception of desirable neurone's outputs of the hidden layer it is necessary to make normalisation of the received meanings, which belong to the segment D. The normalisation will be carried out for each neurone by the following rule.

1) The pattern is determined, for which the value M of the desirable output for the given neurone is the maximal on the module

$$M = \max_k |x^k| = \overline{1,L}, \qquad (22)$$

where L – the number of patterns.

2) If this value belongs to segment D, the procedure is finished.

3) The desirable outputs for the given neurone for all patterns are recalculated according to the formula

$$\overline{x^k} = \frac{x^k}{M}, k = \overline{1,L} \qquad (23)$$

4) The weights from the given neurone to neurones of the following layer change according to the formula

$$\omega_{ij}^{*} = \omega_{ij} \cdot M, j = \overline{1,J}, \qquad (24)$$

where i – the number of considered neurones, J – the quantity of neurones in the following layer.

3. Algorithm

The algorithm consist of the following steps:

1. The random initial of weights is made and the minimal root-mean-square error of the network E_m is set.
2. The modification of weights and thresholds according to expressions (9) and (10) only for the last layer is made. Simultaneously with each pattern according to (21) the desirable outputs $\overline{x_i}$ of neural elements from the hidden layer are defined.
3. Outputs of hidden layers of network are considered only: as the entrance information the outputs $\overline{x_i}$ of neural elements are used, as target - reference outputs
   - 3.1. The modification of weights, thresholds and desirable outputs according expressions (9), (10) and (21) accordingly is made for L entrance patterns.
   - 3.2. Item 3.1 is repeated, while the total root-mean-square error of the examined fragment of the neural network will not become less than E_m.
4. The desirable outputs are normalized according to formulas (22)-(24).
5. The modification on L patterns of weights and thresholds of the following layer of the network is made. Thus the error i-th of the neural element is equal to $\gamma_i = x_i - \overline{x_i}$.
6. The procedure is repeated from step 2, while the total root-mean-square error of the neural network will not become less than E_m.

4. Testing

The experimental check of the received results was carried out on the task of coding of the information. Thus the neural network should on the basis of cyclic coding carry out transformation informational polynomial in a surplus code.

Let word length of informational polynomial be n = 4, and superfluous polynomial m = 7. Then the architecture of the multilayer neural network contains 4 input and 7 output neurones. In the hidden layer we shall use 8 neural elements. Then we have the neural network consisting of 3 layers the with size of training set L = 16.

The experiments have shown that in comparison with other algorithms 100 % stability is got. So with any initialisation of weights the neural network was trained up to the minimal error. With the use of standard backpropagation only in 10 % of all attempts to train the network the acceptable result was reached.

5. Conclusion

The experimental check of algorithm is carried out on the basis of the developed method. It is characterised by independent training of each of layers of a neural network. The experiments have shown that the offered algorithm has greater stability in relation to standard backpropagation. In the offered method the potential opportunities for automatic generation of neural network architecture are incorporated also. Nowadays in this direction the researches will be carried out.

6. References

1. Silra F., Almeida L. Speeding up backpropogation.// Advanced neural computers, North-Holland, pp.151-160, 1990.
2. Saarinen S., Bramley R., Cybenko G. Ill-conditioning in neural network training problems. SIAM Journal on Scientific Computing, 14:693-714, 1993.
3. Golovko V., Savitsky Ju., Gladischuk V. A neural net for predicting problem. Timisoara: University of Timisoara, Romania, 1996.

Determination of Features of the Dependence of Neuron Net Output Coordinates on Other Coordinates and Parameters of Neuron Net

Vladimir Ptitchkin
Byelorussian State University of Informatics and Radioengineering

ABSTRACT

The paper presents the deduction of main relationships of back-propagation algorithm on the basis of approximation of the unknown dependence of the efficiency factor on neural net parameters by the linear part of Taylor series. Such an approach for deduction of known results is intended to relax the limits of application of the back-propagation algorithm and to eliminate some of its shortcomings on the basis of new methods of approximation of the before-mentioned dependence. As an example, this paper examines the possibility of application of back-propagation algorithm in the case of non-differentiable activation functions.

1: Introduction

The back-propagation algorithm is of great importance for the theory and practice of neural nets. It is the algorithm for learning of multilayer neural nets. It is based on the iterative procedure of finding of minimal mean-square error in the parameters space using the method of quickest descent [1].

In general, the problem of determination of new values of the adjustable parameters of neural net can be considered as the problem of the approximate expression of the dependence of the efficiency factor of the neural net on before-mentioned parameters, and also as a problem of application of some characteristics of this dependence for parameters correction. We shall understand the approximate determination of the dependence as one of the forms of its approximations, for example, a finite part of series; let's understand the characteristics of this dependence as the coefficients of series. From this point of view, the traditional back-propagation algorithm can be considered as the application of coefficients of approximate description of the before-mentioned dependence in the form of Taylor series linear part, because the coefficients of Taylor series are the parameter derivatives of the output coordinate.

We think that many shortcomings of the back-propagation algorithm are the results of application of activation function derivatives in the observation points. Some of these shortcomings can be eliminated by application of another approaches for the approximation of the unknown dependence of the efficiency factor on net parameters. For example, the application of statistical linearization method allows to deduct the linear approximation of dependence between coordinates, without taking into consideration the differentiability of activation functions [2]. In such conditions, it is naturally enough to return to the application of sign-functions as activation functions.

However, the attempt not to use the derivatives of neural net non-linear characteristics leads the loss of theoretical ground of back-propagation algorithm in the form of the method of quickest descent. For example, applying the method of statistical linearization for approximate description of the dependence of neural net output coordinates on any other coordinates, as original approach for neural nets analysis, we have not found any groung for development of this procedure for net learning [2]. From the heuristical point of view, the non-essential variations in the procedure of deduction of main relationships for back-propagation algorithm (in comparison with Taylor series approximation) must not lead to the impossibility to apply the linear approximations in such a manner as in back-propagation algorithm.

There are no doubts that the application of approximation-based considerations for deduction of main relationships for back-propagation algorithm (using Taylor series as a tool for their deduction) can give, as a result, only well-known relationships [1].

However, it's interesting to obtain these relations in just such a way. Such result will allow to generalize and adjust the above-mentioned procedure for specific conditions. This is the main goal of the paper proposed. The additional result is the obtaining of the simplified (from the engineering point of view) procedure of deducting of main relationships for back-propagation algorithm; this procedure, as we think, can be used for training specialists.

2: Matrix description of neural net

Every neuron, neural layer and neural net (NN) as a whole can be descripted by sets of states, input coordinates and output coordinates, which are represented by vectors S, U, and Y respectively. There function in a steady-state mode can be descripted by two equations: transition equation $S = A(U)$ and output equation $Y = F(S)$.

The formal neuron can be represented as a multi-input summator and non-linear converter, which are connected serially. Therefore, transition equation can be denoted as summator equation, and states can be denoted as summator outputs. Let's denote the above-introduced coordinates U, S, and Y as summator inputs, summator outputs, and non-linear elements outputs, respectively. The summator equations are linear for the single neuron, as well as for the neural layer.

The summator equations can be represented in matrix form: $S = B + AU$, where B is a column of free terms. The elements of this column are the threshold values of formal neurons, only with the inverse sign. The matrix A, in turn, contains the synaptic weights. The threshold values and the synaptic weights are the parameters of NN. There is one particular feature of conversions implemented by NN: the non-linear conversion of each state is independent from the conversions of any other state. It means that the dimensions of vectors Y and S are equal, and also, each coordinate of Y is functionally dependent on the respective coordinate of S. In another words, the vector equation $Y = F(S)$ can be considered as a complex of one-dimensional functions $y_i = f_i(s_i)$. For homogeneous NN, the activation functions for all neurons are the same, i.e. $f_i = f$. For simplicity, we shall consider only homogeneous NN with serial connections.

Let's suppose that all neurons are numbered in such a way that when the output of i-th neuron is calculated, all inputs of this neuron are already calculated. Such a condition determines the separation of neurons into layers. The collating of neurons in the net allows to collate net coordinates, and the separation of neurons into layers allows to represent the vectors in a form of blocks. Let's denote

$$Y^0 = (Y_0, Y)', Y = (Y_1, Y_2, ..., Y_m)',$$
$$S^0 = (S_0, S)', S = (S_1, S_2, ..., S_m)',$$

where Y_j, and S_j are the vector of input coordinates and the vector of states for j-th layer, respectively. Some uncertainty in separation of coordinates and neurons into layers is introduced by sensor elements. Let's consider them as neurons of zero layer. In NN, the inputs of any neuron are the outputs of some other neurons; therefore, the coordinates of vector U can be excluded from the set of equations representing the NN. Such an exclusion is especially simple after the separation of neurons into layers and collation of output coordinates into ranks. The vector of states for any layer of NN with serial connections can be determined through the vector of outputs of the preceding layer:

$$S_i = B_i + A_i Y_{i-1}, (i = 1, 2, ..., m). \quad (1)$$

If, as a result of the linearization, the connection between coordinates y_i and s_i is expressed as

$$y_i = q_i + k_i s_i, \quad (2)$$

then the set of such equations can be represented in a matrix form for NN, as well as for any particular layer:

$$Y_i = Q_i + K_i S_i, (i = 1, 2, ..., m), \quad (3)$$

where Q_i is a column of free terms, K_i is a diagonal matrix of linearization coefficients for activation functions.

In accordance with our goal, we shall apply the expansion into Taylor series for linearization. Assume that there is a signal $Y_0 = Y^*_0$ on the input of NN. Let's denote by asterisk (*) the values of all NN coordinates determined by this input signal. From (1), we shall obtain:

$$S^*_i = B_i + A_i Y^*_{i-1}, S_i - S^*_i = A_i (Y_{i-1} - Y^*_{i-1}). \quad (4)$$

Let's represent deviation equations in the following form:

$${}^0S_i = A_i {}^0Y_{i-1}, \quad (5)$$

where

$${}^0S_i = S_i - S^*_i, {}^0Y_{i-1} = Y_{i-1} - Y^*_{i-1}$$

Now we shall start the linearization of non-linear equation $y_i = f_i(s_i)$. Assuming that the functions f_i are differentiable everywhere in their domains, let's apply for them the expansion into Taylor series in the vicinity of $s_i = s^*_i$. Then, instead of the general expression (2), we can use more specific and convenient expression:

$$y_i - y^*_i = k^*_i (s_i - s^*_i) \text{ or } {}^0y_i = k^*_i {}^0s_i, \quad (6)$$

where

$${}^0y_i = y_i - y^*_i, \; {}^0s_i = s_i - s^*_i, \; k^*_i = f_i'(s^*_i),$$

i.e. $k^*_i = f_i'(s^*_i)$ is a derivative of i-th activation function in a "point" of the corresponding value of input coordinate. Therefore, the vector of output coordinates of i-th layer also can be expressed in matrix form:

$${}^0Y_i = K^*_i {}^0S_i, \quad (7)$$

where K^*_i is a diagonal matrix of linearization coefficients, i.e. the derivatives in the "point" under analysis.

From (5) and (7), we can obtain:

$${}^0Y_i = K^*_i A_i {}^0Y_{i-1} \; (i = 1, 2, ..., m). \quad (8)$$

This relationship is an approximate expression of dependence of i-th layer output vector on the output vector of preceding layer or (the same) the dependence on the input vector of the i-th layer. However, this approximate relationship allows to obtain the exact expression for the derivation of one coordinate with respect to another. This possibility is guaranteed by the application of just Taylor series for function approximation.

3: Determination of linearized relationships between net coordinates

It's not difficult to obtain the linearized expression of dependence of states or outputs vector of i-th layer on the states or outputs vector of (i-2)-th layer, etc. For this investigation, the output vector of the last layer is of special interest. For obtaining the derivative in the particular point of NN output coordinates (or, the same, in the particular point of output coordinate of last m-th layer) ${}^0Y_m$ with respect to the output coordinate of j-th layer ($0 \le j < m$) we need to apply the expression (8) k times, where $k = m-j$:

$${}^0Y_m = (K^*_m A_m)(K^*_{m-1} A_{m-1}) \ldots (K^*_{m-k} A_{m-k}) {}^0Y_{m-k-1}. \quad (9)$$

The parentheses in this expression are set for better readability only, because the operations of multiplication may be performed in any order. However, such a form of expression facilitates the correction of error in the case of missing some intermediate term.

Applying the expression (5) with $i = m-k$

$${}^0S_{m-k} = A_{m-k} {}^0Y_{m-k-1}$$

we can obtain the expression of linear dependence of NN output coordinate on the state of j-th layer:

$${}^0Y_m = (K^*_m A_m)(K^*_{m-1} A_{m-1}) \ldots (K^*_{m-k+1} A_{m-k+1}) K^*_{m-k} {}^0S_{m-k}. \quad (10)$$

From (5) and (7) we can obtain that

$${}^0S_i = A_i K^*_{i-1} {}^0S_{i-1}. \quad (11)$$

From here, we can deduct the following expressions:

$${}^0S_m = A_m(K^*_{m-1} A_{m-1}) \ldots (K^*_{m-k} A_{m-k}) {}^0Y_{m-k-1}, \quad (12)$$

$${}^0S_m = A_m(K^*_{m-1} A_{m-1}) \ldots (K^*_{m-k+1} A_{m-k+1}) K^*_{m-k} {}^0S_{m-k}. \quad (13)$$

Thus, we can obtain the derivative of any coordinate (output or state) for i-th layer with respect to any coordinate of the preceding layer. The derivatives are required for determination of efficiency factor gradient; therefore, it's necessary to determine the expressions of output vector derivatives with respect to inputs of summators of all layers. Thus, the most interesting result for this investigation is the expression (10).

If the determination of output coordinates derivatives was the only goal of NN analysis, the expression (10) would be the final result making the following discussions unnecessary. However, it seems unreasonable to calculate derivatives for every particular layer in order to determine the derivatives for all layers, because the same set of calculations must be repeated for each layer. In such conditions, it seems naturally enough to use the results of derivatives calculation for i-th layer for calculation of derivatives for (i-1)-th layer. Of course, such a method gives real benefit only when the number of layers and/or neurons is great enough. Possibly, under small number of layers and neurons, it is more reasonable to perform more calculations using simple algorithm, than to reduce the

number of arithmetic operations at the expense of complicating the algorithm. Even if we shall not take into consideration the difficulty of understanding the back-propagation algorithm in the recurrent form, just the possibility to eliminate the recurrence seems especially important for the following generalizations.

Thus, in conditions when the number of neurons is not very great, the recurrent implementation of back-propagation algorithm is not of great importance. However, we shall consider this question, first of all, in order to prove that the proposed scheme of investigation leads (from a particular point of view), as a final result, to the back-propagation algorithm.

4: The recurrent form of dependencies

Let's present the expression (10) for two specific cases ($k = 0$, $k = m-1$)

$$ {}^0Y_m = K^*_m {}^0S_m , \tag{14} $$

$$ {}^0Y_m = (K^*_m A_m)(K^*_{m-1}A_{m-1}) \dots (K^*_2 A_2)K^*_1 {}^0S_1 . \tag{15} $$

Let's present the NN output vector ${}^0Y_m$ as a function of state vector ${}^0S_j$ in matrix form:

$$ {}^0Y_m = \Delta_j {}^0S_j . \tag{16} $$

Thus, we can consider the matrix Δ_j as the matrix of output vector derivatives with respect to the states of i-th layer. It's obvious that

$$ \Delta_m = K^*_m . \tag{17} $$

From (10) under k=1, taking into consideration (16)

$$ {}^0Y_m = (K^*_m A_m)K^*_{m-1} {}^0S_{m-1} = \Delta_m A_m K^*_{m-1} {}^0S_{m-1} = $$
$$ = \Delta_{m-1} {}^0S_{m-1} $$

we can obtain:

$$ \Delta_{m-1} = \Delta_m A_m K^*_{m-1} . \tag{18} $$

In turn, under k=2, taking (15) and (18) into consideration

$$ {}^0Y_m = (K^*_m A_m)(K^*_{m-1}A_{m-1})K^*_{m-2} {}^0S_{m-2} = $$
$$ = \Delta_{m-1} A_{m-1} K^*_{m-2} {}^0S_{m-2} = \Delta_{m-2} {}^0S_{m-2} $$

we can obtain

$$ \Delta_{m-2} = \Delta_{m-1} A_{m-1} K^*_{m-2} . \tag{19} $$

Now we can deduce the generalized expression

$$ \Delta_{m-k-1} = \Delta_{m-k} A_{m-k} K^*_{m-k-1} , \tag{20} $$

which can be considered as recurrent equation with initial conditions (17).

This recurrent expression coincides with the main relationship of back-propagation algorithm (up to the designations). The deduction of this expression seems natural if we use matrix terms. In accordance with this algorithm, the calculations are performed in two "directions". The forward-direction calculations include calculation of all NN coordinates in the order of layer number increment, starting from the values of input coordinates. After the determination of output coordinates for the last layer, the reverse-direction calculations are performed, which include calculations of "generalized" errors for the last layer (i.e. the components of vector Δ_m, and vectors Δ_i for all preceding layers, in the order of layer number decrement.

It must be noted that if we use the direct expressions of linear equations of connection between NN coordinates, then the division of calculations into two stages (forward-direction and reverse-direction) will be unnatural. Indeed, the forward-direction calculation of NN coordinates, from layer to layer, consist in sequential calculation of coordinates (i.e. output and state) for each layer, starting from the first layer, i.e. in calculation of Y_0, S_1, Y_1, S_2, Y_2, ..., S_m, Y_m.

After the calculation of the column Y_1, it's possible to determine analytical expressions of dependence of output Y_i on all columns of states S_j under $j \le i$. It's obvious that under i=1, i.e. on the first step of the sequential processing of layers, we can obtain only one dependence:

$$ {}^0Y_1 = K^*_1 {}^0S_1 , \tag{21} $$

However, on he second step we can obtain two dependencies:

$$ \begin{aligned} {}^0Y_2 &= K^*_2 {}^0S_2 , \\ {}^0Y_2 &= K^*_2 A_2 K^*_1 {}^0S_1 , \end{aligned} \tag{22} $$

and on the third step there are three dependencies:

$$ \begin{aligned} {}^0Y_3 &= K^*_3 {}^0S_3 , \\ {}^0Y_3 &= K^*_3 A_3 K^*_2 {}^0S_2 , \\ {}^0Y_3 &= K^*_3 A_3 K^*_2 A_2 K^*_1 {}^0S_1 \end{aligned} \tag{23} $$

and so on.

To make obvious the recurrent nature of the proposed algorithm, let's introduce the designation Δ_{ij} for matrix of connections between vectors Y_i and S_j. Using this matrix, we can express the dependence of the first of the above-mentioned vectors from the second in the following form:

$$ {}^0Y_i = \Delta_{ij}\, {}^0S_j \quad (j \le i). \tag{24} $$

Using this designation, let's present (21) in the following form:

$$ {}^0Y_1 = \Delta_{11}\, {}^0S_1, \tag{25} $$

where $\Delta_{11} = K^*_1$; (22) can be expressed as follows:

$$ {}^0Y_2 = K^*_2\, {}^0S_2 = \Delta_{22}\, {}^0S_2, $$
$$ {}^0Y_2 = K^*_2 A_2 K^*_1\, {}^0S_1 = K^*_2 A_2 \Delta_{11}\, {}^0S_1 = \Delta_{21}\, {}^0S_1, \tag{26} $$

where $\Delta_{22} = K^*_2$, $\Delta_{21} = K^*_2 A_2 \Delta_{11}$; and (23) can be expressed in the following form:

$$ {}^0Y_3 = K^*_3\, {}^0S_3 = \Delta_{33}\, {}^0S_3, $$
$$ {}^0Y_3 = K^*_3 A_3 K^*_2\, {}^0S_2 = K^*_3 A_3 \Delta_{22}\, {}^0S_2 = \Delta_{32}\, {}^0S_2, $$
$$ {}^0Y_3 = K^*_3 A_3 K^*_2 A_2 K^*_1\, {}^0S_1 = K^*_3 A_3 \Delta_{21}\, {}^0S_1 = \Delta_{31}\, {}^0S_1, $$

where

$$ \Delta_{33} = K^*_3, \quad \Delta_{32} = K^*_3 A_3 \Delta_{22}, \quad \Delta_{31} = K^*_3 A_3 \Delta_{21}. $$

In general, we can conclude that after processing of i-th layer ($i \ge 1$) we have i matrices Δ_{ij} ($1 \le j \le i$) calculated. For next, (i+1)th layer, we can calculate i+1 matrices in accordance with the following rule:

$$ \Delta_{(i+1)j} = K^*_{(i+1)} A_{(i+1)}\, \Delta_{ij}, \ (1 \le j \le i) $$
$$ \Delta_{(i+1)(i+1)} = K^*_{(i+1)}. $$

After processing of the last (m-th) layer we have m matrices Δ_{mj} ($1 \le j \le m$) calculated. They are used as the above-mentioned matrices Δ_j. Indeed, the meaning of these matrices is determined by connection equation (16); after its comparison with (24), we can see that $\Delta_{mj} = \Delta_j$.

Thus, we can conclude that there are at least three methods that can be applied for calculation of the linearized dependence of NN output coordinate on the states of each layer ("in the vicinity of equilibrium position"). The first method consists of direct calculation of each of the dependencies in the matrix form. The second method includes two stages: 1) the calculation of "equilibrium position" for all coordinates of each layer (the forward-direction calculations) and 2) the calculation of dependence of the last layer output coordinates (i.e. the output coordinates of NN) on the states of preceding layers (the reverse-direction calculation). In the third method, it is proposed to determine the relationship for output coordinate in parallel with the calculation of "equilibrium position", i.e. the values of NN coordinates determined by the value of the input coordinates.

5: Determination of output coordinates derivatives with respect to parameters

In the back-propagation algorithm, it's necessary to calculate the derivatives of output coordinates with respect to parameters (not with respect to the states of layers). While the relation between these derivatives is rather simple, let's present some relationships between them. First, the matrix Δ_j between the centralized values ${}^0Y_m = \Delta_j\, {}^0S_j$ can be considered as the matrix of derivatives of non-linear dependence of Y_m on S_j with $S_j = S^*_j$. Such an assertion is grounded on application of Taylor series for approximation of all non-linear functions used in this conversion. Each element of matrix $\Delta_j = (\delta_{j\,pq})$ is the derivative of p-th component of NN output vector (m-th layer) with respect to q-th component of states vector of j-th neuron layer $s_{j\,q}$.

$$ \delta_{j\,pq} = \partial y_{m,\,p} / \partial s_{j,\,q}. $$

If given the matrix expression of dependence of j-th states column S_j on output vector of the preceding layer Y_{j-1} (see (1))

$$ S_j = B_j + A_j Y_{j-1} $$

we take only the q-th line

$$ s_{j,\,q} = b_{j,\,q} + a_{j,\,q,\,1} y_{j-1,\,q,\,1} + a_{j,\,q,\,2} y_{j-1,\,q,\,2} + \dots, $$

where $b_{j\,q} = -\theta_{j\,q}$, then it is obvious that

$$ \partial y_{m,\,p} / \partial b_{j,\,q} = \partial y_{m,\,p} / \partial s_{j,\,q} = \delta_{j\,pq}, $$

$$ \partial y_{m,\,p} / \partial a_{j,\,q,\,k} = \partial y_{m,\,p} / \partial s_{j,\,q} \times \partial s_{j,\,q} / \partial a_{j,\,q,\,k} = $$
$$ = \delta_{j\,pq} \times y_{j-1,\,q,\,k}. \tag{27} $$

Thus, after determination of the derivatives of output column with respect to the states of all NN layers, we can determine the derivatives of output coordinates with respect to all NN parameters. Of course, for coordinate values we take their values in the experiment

under consideration. In turn, on the basis of output coordinate derivative with respect to the parameters, we can determine the derivatives of the square error with respect to the same parameters. In fact, this is the end of the deduction of main relationships of back-propagation algorithm.

It's not difficult to see that the relationships obtained coincide (up to the designations) with the well-known derivative expressions for back-propagation algorithm. The main difference is not in final results but in the way by which these results are obtained. In classical presentation of the back-propagation algorithm, the problem under consideration is the determination of derivatives of composite function (i.e. mean-square error) as such, without any connection with the problem of transfer function approximation by power polynomials. In such conditions, the application of chain rule and recurrent-type calculations for determination of composite functions derivatives seems natural and even necessary. However, the application of derivatives in any case implies the approximation of functions in the vicinity of some point. In the proposed approach, the starting point for finding better parameters values (in comparison with the values used in the experiments) is the approximation of the dependence of efficiency factor on parameters in the range of variables values in the training set. We think that in such conditions the application of derivatives is not a matter of principle, but providing they are used, the proposed algorithm gives the same results as the classical approach for back-propagation algorithm. This was the main goal of the investigation proposed, and we think that this goal is achieved.

6: Determination of linearized relationships without application of derivatives

As a rule, the practical implementation of back-propagation algorithm includes some violations of the assumptions used for proving of algorithm convergency. These violations include not only the selection of the final step, but also the correction of parameter values after the presentation of each pattern (the pulse method). At the same time, all considerations are valid only provided that the values of parameters remain the same during the presentation of all the patterns of the training set. Although the pulse method gives some benefits in many cases, it requires additional investigations. To simplify the following discussions with respect to the possibility of NN learning without the determination of derivatives, let's assume that the new values of parameters for back-propagation algorithm are implemented only after the end of processing of the training set.

In such conditions, the determination of "derivatives" of efficiency factor with respect to the NN parameters without the differentiation of activation functions can be considered as the determination of coefficients of linear regression of efficiency factor on the outputs of summators. These coefficient can be determined, for example, using the method of least squares. These coefficients may be considered as the "generalizer errors" of back-propagation algorithm. However, the calculation of the derivatives of efficiency factor with respect to the parameters in accordance with (27) is strictly grounded on the basis of differential calculus; at the same time, the application of (27) in the case under consideration requires strict grounds. As a heuristical recommendation, it may be proposed to apply the mean value of output coordinate instead of the value taken from the particular experiment.

To investigate the acceptability of the proposed recommendation for practical purposes, the numerical experiments, as a minimum, are necessary. As a test example for such experiments, we can use two NNs, the only difference between which is the activation function of formal neurons. For one of these NNs, the sign-function must be used as the activation function. For another NN, the sigmoid-function must be used with such a great value of the parameter that makes this function practically indistinguishable from the sign-function. It's obvious that, providing the beforementioned heuristical recommendation is valid, the values of gradient of the efficiency factor for such NNs must be approximately the same. Such numerical experiment was carried out for one test example, and the results were satisfactory. Of course, the single example is not adequate ground to recommend the method for wide practical application; however, we think that the main goal of this investigation is achieved. This goal was to demonstrate the opportunities of back-propagation algorithm, without respect to its particular implementation.

References

[1] Rumelhart D.E., Hinton G.E., Williams R.J. Learning internal representations by error propagation. In Parallel distributed processing, vol. I, Cambridge, MA: MIT Press, 1986.

[2] Птичкин В.А. Анализ нейросетевых преобразований линеаризационными методами. // Нейрокомпьютер, М., 1999

Methods of Partial Logic

Anatoly Prihozhy

*State Polytechnical Academy, 65 Block 11a, F.Skorina ave., Minsk 220027 Belarus
Fax: 375-017 2327153, e-mail: aprihozhy@bspa.unibel.by

Abstract

This paper presents novel theoretical results obtained in the field of partial logic. New operations (including a very useful minimization operation), laws, and expansions are introduced. Traditional Boolean function representation forms for the completely specified functions are generalized for the incompletely specified and partial functions.

1. Introduction

Most of the existing logic level synthesis techniques are based on the traditional two-valued logic [1]. The logic allows generation of various representations for the same logic function that can be mapped to digital circuits with different parameters. The most significant representations are as follows:

- sum of products
- product of sums
- Reed-Muller expressions
- decision diagrams.

Two key types of decision diagrams are used [1,2,4]:

- binary decision diagrams (BDDs) derived from the Shannon expansion
- functional decision diagrams (FDDs) derived from the positive and negative Davio expansions.

The ROBDDs (reduced ordered BDDs) [2,4] being a Boolean function canonical representation form, are the most popular type of decision diagrams widely used for efficient modeling, synthesis, and verification of digital circuits.

High-level synthesis systems [5-6] need efficient logic optimization techniques.

Incompletely specified functions are a very useful formalism for generating different alternatives during logic optimization process [3].

This paper presents novel theoretical results in the field of logic that uses three values: true, false, and don't care. New operations and representation forms (expressions) for the three-valued functions, laws and decomposition types in the logic are described in the paper.

2. Partial and incompletely specified variables and functions

The traditional total logic considers two values: *true (1)* and *false (0)*. The partial logic considers three values: *true (1)*, *false (0)*, and *don't care* (*dc* or –). The *don't care* value can be replaced with *true* or *false* arbitrarily. A total function $f(x_1,...,x_n)$ is a mapping $f: B^n \rightarrow B$ where $B=\{0,1\}$. A partial function $g(y_1,...,y_m)$ is a mapping $g: M^m \rightarrow M$ where $M=\{0,1,-\}$. An incompletely specified function $h(x_1,...,x_n)$ is a mapping $h: B^n \rightarrow M$. A variable which takes values from the set B will be called a total variable, and a variable which takes values from the set M will be called a partial variable.

A Value-Domain Representation (VDR) is the following encoding of a partial variable y_i with a pair $(v_i|d_i)$ of total variables:

$$y_i = \begin{cases} 0, & \text{if } v_i=0 \text{ and } d_i=1, \\ 1, & \text{if } v_i=1 \text{ and } d_i=1, \\ dc, & \text{if } v_i \in \{0,1\} \text{ and } d_i=0. \end{cases}$$

The variable v_i is called a value variable and the variable d_i is called a domain variable.

Due to VDR a partial function $z=g(y)$ of m three-valued arguments is represented by an incompletely specified function $(v|d)=g'((v_1|d_1), ...,(v_n|d_n))$ of $2m$ two-valued arguments with on-set $g^{on}=(v\&d)^{on}$, *off-set* $g^{off}=(\sim v\&d)^{on}$, and *don't-care-set* $g^{dc}=(\sim d)^{on}$.

Monadic and dyadic partial operations are transformed to operations on pairs of total functions (Table 1). The operations are used for construction of partial logic expressions and mixed total-partial logic expressions. Pairs of the expressions representing the same partial function constitute partial logic laws. The laws which transform a partial expression to a pair of total expressions, connect the partial logic to the traditional total logic. They include:

$$\sim(v_1|d_1) = (\sim v_1|d_1).$$
$$(v_1|d_1)\&(v_2|d_2) = (v_1\&v_2|d_1\&d_2+\sim v_1\&d_1+\sim v_2\&d_2),$$
$$(v_1|d_1)+(v_2|d_2) = (v_1+v_2|d_1\&d_2+v_1\&d_1+v_2\&d_2),$$
$$(v_1|d_1)\rightarrow(v_2|d_2) = (v_1\rightarrow v_2|d_1\&d_2+\sim v_1\&d_1+v_2\&d_2),$$
$$(v_1|d_1)\oplus(v_2|d_2) = (v_1\oplus v_2|d_1\&d_2).$$

The left parts of equalities contain partial operations including negation (~), conjunction (&), disjunction (+), implication (→), exclusive OR (⊕), and their right parts contain total operations with the same notations.

Table 1

Partial logic operations

N	Operation name	Values: 0 1 – 0 1 – 0 1 – / 0 0 0 1 1 1 – – –	Notation
1	Negation	1 0 –	$\sim(v_1\|d_1)$
2	Conjunct.	0 0 0 0 1 – 0 – –	$(v_1\|d_1)\&(v_2\|d_2)$
3	Disjunct.	0 1 – 1 1 1 – 1 –	$(v_1\|d_1)+(v_2\|d_2)$
4	Implicat.	1 0 – 1 1 1 1 – –	$(v_1\|d_1)\rightarrow(v_2\|d_2)$
5	Excl. OR	0 1 – 1 0 – – – –	$(v_1\|d_1)\oplus(v_2\|d_2)$

3. Minimization operation

In the pair $(v|d)$ function d is fixed. The function v can be replaced with another total function v_t such that

$$(v_t|d) = (v|d)$$

or

$$v_t\&d = v\&d.$$

In other words, VDRs $(v_t|d)$ and $(v|d)$ represent the same incompletely specified function. If V is the set of functions v_t then for each $v_t \in V$ the inequality

$$(v\&d)^{on} \subseteq v_t^{on} \subseteq (v+\sim d)^{on}$$

holds.

A minimization operation $min(v|d)$ is a mapping $min: F\times F \rightarrow F$ where F is the set of total functions $f: B^n \rightarrow B$ In fact, the operation selects one function from the set V. Various definitions for $min(v|d)$ are possible. They depend on which representation forms for v and d are used. In work [7] a definition of the operation on BDDs and in particular on ROBDDs is given. The operation allows decrease in the number of nodes and edges in the BDD v as shown in Fig.1.

Figure 1: Minimization operation on BDDs, an example

4. Partial logic laws

The following laws in the partial logic generalize known laws in the traditional logic:

$(v|d) = \sim\sim(v|d),$
$(v_1|d_1)\&(v_2|d_2) = (v_2|d_2)\&(v_1|d_1),$
$(v_1|d_1)\&((v_2|d_2)\&(v_3|d_3)) =$
$\quad((v_1|d_1)\&(v_2|d_2))\&(v_3|d_3),$
$(v_1|d_1)+(v_2|d_2) = (v_2|d_2)+(v_1|d_1),$
$(v_1|d_1)+((v_2|d_2)+(v_3|d_3)) =$
$\quad((v_1|d_1)+(v_2|d_2))+(v_3|d_3),$
$(v_1|d_1)\&((v_2|d_2)+(v_3|d_3)) =$
$\quad((v_1|d_1)\&(v_2|d_2))+((v_1|d_1)\&(v_3|d_3)),$
$(v_1|d_1)+((v_2|d_2)\&(v_3|d_3)) =$
$\quad((v_1|d_1)+(v_2|d_2))\&((v_1|d_1)+(v_3|d_3)),$
$\sim((v_1|d_1)\&(v_2|d_2)) = \sim(v_1|d_1)+\sim(v_2|d_2),$
$\sim((v_1|d_1)+(v_2|d_2)) = \sim(v_1|d_1)\&\sim(v_2|d_2),$
$(v_1|d_1)\&((v_1|d_1)+(v_2|d_2)) = v_1|d_1,$
$(v_1|d_1)+((v_1|d_1)\&(v_2|d_2)) = v_1|d_1,$
$(v_1|d_1)\text{->}(v_2|d_2) = \sim(v_1|d_1)+(v_2|d_2),$
$(v|d)\&(v|d) = v|d,$
$(v|d)+(v|d) = v|d,$
$(v|d)\&\sim(v|d) = 0|d,$
$(v|d)+\sim(v|d) = 1|d,$
$(v|d)\&(1|1) = v|d,$
$(v|d)+(0|1) = v|d,$
$(v|d)\&(0|1) = 0|1,$
$(v|d)+(1|1) = 1|1,$
$(0|1)\text{->}(v|d) = 0|1,$
$(v|d)\text{->}(v|d) = 1|d,$
$(v|d)\text{->}(1|1) = 1|1,$
$(v_1|d_1)\text{->}(v_2|d_2) = \sim(v_2|d_2)\text{->} \sim(v_1|d_1).$

New laws of the partial logic are as follows:

$(v_1|d)\&(v_2|d) = v_1\&v_2|d,$
$(v_1|d)+(v_2|d) = v_1+v_2|d,$
$(v|d_1)\&(v|d_2) = v|d_1\&d_2+\sim v\&(d_1+d_2),$
$(v|d_1)+(v|d_2) = v|d_1\&d_2+v\&(d_1+d_2),$
$v|v = 1|v,$

$\sim v|v = 0|v,$
$v\&d|d = v|d,$
$v+d|d = 1|d,$
$v\&\sim d|d = 0|d,$
$v+\sim d|d = v|d,$
$v|v\&d = 1|v\&d,$
$v|\sim v\&d = 0|\sim v\&d,$
$v|v\&d = (v|d)+(v|0),$
$v|\sim v\&d = (v|d)\&(v|0),$
$v|v+d = (v|1)+(0|d),$
$v|\sim v+d = (v|1)\&(1|d),$
$\sim v|v+d = (\sim v|1)+(1|d),$
$\sim v|\sim v+d = (\sim v|1)\&(0|d),$
$f|x_i = f(x_i=1)|x_i,$
$f|\sim x_i = f(x_i=0)|x_i,$
$v|v\oplus d = \sim d|v\oplus d,$
$v|v\equiv d = d|v\equiv d,$
$\sim v|v\equiv d = \sim d|v\equiv d,$
$f(v)|v\oplus d = f(\sim d)|v\oplus d,$
$f(v)|v\equiv d = f(d)|v\equiv d,$
$min(\sim v|d) = \sim min(v|d),$
$v\&d \le min(v|d) \le v+\sim d.$

5. Partial logic expansions

The work [1] proofs that only the Shannon and Davio expansions are useful for representation and manipulation of functions in the total logic. The Shannon expansion of function $f(x)$ in the logic is as follows:

$$f(x) = x_i \& f_{xi=1} + \sim x_i \& f_{xi=0}$$

where x_i is a total variable, $f_{xi=1}$ is a cofactor of function $f(x)$ on $x_i=1$ and $f_{xi=0}$ is a cofactor of the function on $x_i=0$.

The following generalization for the Shannon expansion holds in the partial logic:

$$f(x) = \alpha(x) \,\&\, min(f(x)|\alpha(x)) + \sim \alpha(x) \,\&\, min(f(x)|\sim\alpha(x))$$

In the expansion variable x_i is replaced with an arbitrary total function $\alpha(x)$ and the cofactors are replaced with the operations minimizing $f(x)$ on $\alpha(x)$ and $\sim\alpha(x)$ respectively. When $\alpha(x)=1$ then

$$1 \,\&\, min(f(x)|\,1) + 0 \,\&\, min(f(x)|\,0) = f(x).$$

When $\alpha(x)=0$ we obtain the same result.

The partial logic expansion

$$f(x) = min(f(x)|\sim\alpha(x)) \oplus \alpha(x) \,\&\, (min((x)f|\alpha(x)) \oplus min(f(x)|\sim\alpha(x)))$$

is a generalization for the positive Davio expansion. A generalization for the negative Davio expansion is derived from the positive one by means of replacement of the function $\alpha(x)$ with its negation $\sim\alpha(x)$.

6. Function representation forms

The table form is the basic one for the representation of the partial and incompletely specified Boolean functions.

The partial Sum of Products is represented by the following expression

$$f = \Big(\sum_{f(a)=1}^{m} \&_{j=1}^{a_j} (y_j\,|1)\Big) + \Big(\sum_{f(a)=0}^{m} \&_{j=1}^{a_j} (0|y_j)\Big) =$$

$$\Big(\sum_{f(a)=1}^{m} \&_{j=1}^{a_i} y_j\,|1\Big) + \Big(0\,|\sum_{f(a)=0}^{m} \&_{j=1}^{a_i} y_j\Big),$$

where $a=(a_1,\ldots,a_m)$ and

$$y_j = \begin{cases} \sim y_j, & \text{if } a_j=0, \\ y_j, & \text{if } a_j=1. \end{cases}$$

if $a_j \in B$, and

$$y_j = \begin{cases} \sim vj \,\&\, dj, & \text{if } a_j=0, \\ y_j \,\&\, dj, & \text{if } a_j=1, \\ \sim d_j, & \text{if } a_j=\text{'-'}. \end{cases}$$

if $a_j \in M$, where v_j, d_j are two-valued variables encoding the three valued variable y_j.

The partial Product of Sums is represented by the following expression

$$f = \Big(\prod_{f(a)=0}^{m} \&_{j=1}^{\sim a_j} (y_j\,|1)\Big) \,\&\, \Big(\prod_{f(a)=-}^{m} \&_{j=1}^{\sim a_j} (1\,|y_j)\Big) =$$

$$\Big(\prod_{f(a)=0}^{m} \&_{j=1}^{\sim a_i} y_i\,|1\Big) \,\&\, \Big(1\,|\prod_{f(a)=-}^{m} \&_{j=1}^{\sim a_i} y_j\Big).$$

7. If-decision diagrams

The if-decision diagrams (IFDs) and functional if-decision diagrams (FIFDs) [7] are constructed though using the generalized Shannon and Davio expansions. Basic fragments of IFDs and FIFDs are shown in Fig.2.

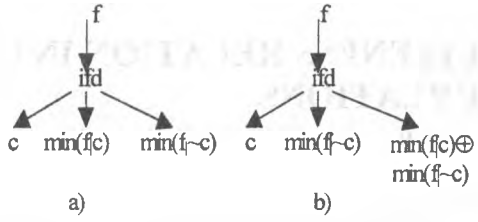
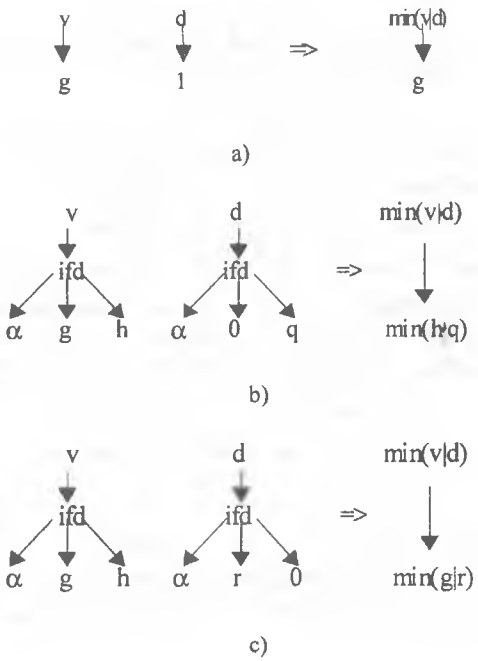


Figure 2: Construction of a) IFD and b) FIFD

The IFD is a generalization for the BDD and the FIFD is a generalization for the FDD. The IFD is represented by a rooted directed noncyclic graph the terminal nodes of which are labeled *0*, *1*, x_i, and $\neg x_i$ and the nonterminal nodes are not labeled and have exactly three successors. The diagram graph is reduced if it does not include identical subgraphs. The IFDs and FIFDs extend the set of representation alternatives and allow a compressed representation and efficient manipulation of Boolean functions.

In order to manipulate the IFDs, we define the minimization operation on this type of decision diagrams. Four cases in Fig.3 define the minimization result for different source IFDs *v* and *d*. As the figure shows, the operation may remove nodes and edges from the diagram *v* in cases b) and c).



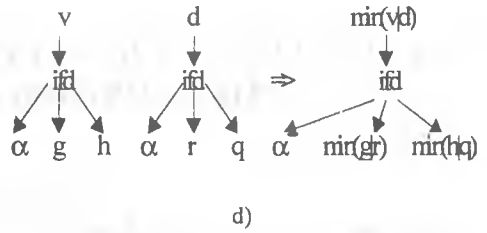


Figure 3: Minimization operation on IFDs

References

[1] B. Becker and R.Drechsler, "How Many Decomposition Types Do We Need?", presented at the ED&TC European Conference, Paris, France, 1995.

[2] R.E.Bryant, "Graph-based algorithms for Boolean function manipulation", IEEE Transactions on Computers, Vol C-35, pp. 677-691, August 1986.

[3] M. Damiani, G. DeMicheli, "Don't Care Set Specification in Combinational and Synchronous Logic Circuits", IEEE Trans. On CAD, vol.12, 1993, pp.365-388.

[4] S. Malik, A. Wang, R. Brayton, A. Sangiovanni-Vincentelli, "Logic Verification Using Binary Decision Diagrams in a Logic Synthesis Environment", in Proceedings of the Int. Conference on CAD, 1988, pp. 6-9.

[5] J.P. Mermet ed., Fundamentals and Standards in Hardware Description Languages. Boston: Kluwer Academic Publishers, 1993.

[6] A. Mozipo, D. Massicotte, P. Quinton, T. Risset, "Automatic Synthesis of a Parallel Architecture for Kalman Filtering using MMAlpha", in Proceedings of the Int. Conference on Parallel Computing in Electrical Engineering, Tech. University of Bialystok, Poland, 1998, pp.201-296.

[7] A. Prihozhy, "If-Diagrams: Theory and Application", in Proceedings of the Conference PATMOS'97, UCL, Belgium, 1997, pp.369-378.

GENERALIZATION OF RUSPINI'S LIKENESS RELATION IN CASES OF SUBNORMAL RELATIONS

Viattchenin D.A.
Institute of Philosophy and Law of the National Academy of Sciences of Belarus
Fax: +375 17 284 2925

Abstract

The note deals in a preliminary way with the problem of a generalization of a likeness relation. Preliminary results are considered. Definition of generalized likeness relation is proposed. Some preliminary conclusions are made.

Keywords: clustering, fuzzy relation, projection.

1. Introduction

A fuzzy approach to clustering is very effectual, because majority relationships between real objects are fuzzy and initial structure of the set of objects is fuzzy also.

One of the fundamental problems in fuzzy cluster analysis is a relationship between the initial structure of the set of objects and its representation in clustering techniques. As a result of the consideration of this problem is a likeness relation, which was proposed by Ruspini (1982). The main purpose of this theoretical note is generalization of the likeness relation for feeble similarity relations cases.

In part 2 of this paper fuzzy similarity relations and some their properties are considered. In part 3 of the paper the likeness relation is described and its generalization are made. In part 4 some concluding remarks and the outlook for the investigations are discussed.

2. Fuzzy similarity relations

The likeness relation was proposed by Ruspini on a basic of fuzzy similarity relation. Let's consider a few kinds of fuzzy similarity relations.

In general, Kaufmann (1975) defined a fuzzy similarity relation S as a binary fuzzy intransitive relation on X, which possesses the symmetric property

$$\mu_S(x,y) = \mu_S(y,x), \forall x, y \in X, \quad (1)$$

and the reflexivity property

$$\mu_S(x,x) = 1, \forall x \in X, \quad (2)$$

where μ_S is a membership function of the fuzzy relation and X is an initial set of elements. We will call this kind of fuzzy similarity relations the usual similarity relation and indicate S_2.

Viattchenin (1997) proposed a feeble similarity relation which we will indicate S_1, and a powerful similarity relation which we will indicate S_3. A feeble similarity relation was defined as a binary fuzzy intransitive relation on X, which possesses the symmetric property (1) and the feeble reflexivity property:

$$\mu_S(x,y) \le \mu_S(x,x), \forall x, y \in X. \quad (3)$$

The feeble similarity relation which possesses the strict inequality condition in (3):

$$\mu_S(x,y) < \mu_S(x,x), \forall x, y \in X \quad (4)$$

is the strict feeble similarity relation. We will indicate this kind of fuzzy similarity relations S_0.

The feeble similarity relation is a subnormal relation, that is $\Pr oj(S_1) < 1$, where $\Pr oj(S_1)$ is the global projection of feeble similarity relation, if strictly restriction condition is met, that was demonstrated by Viattchenin (1998). A condition

$$\mu_{S_1}(x,x) < 1, \forall x \in X \quad (5)$$

is strictly restriction condition for a feeble similarity relation.

A powerful similarity relation was defined as a binary fuzzy intransitive relation on X, which possesses the symmetric property (1) and a powerful reflexivity property. A fuzzy relation possesses the powerful reflexivity property, if the condition

$$\mu_S(x,y) < 1, \forall x, y \in X, x \ne y \quad (6)$$

is met together with condition (2).

From definitions of fuzzy similarity relations a next condition follows: if S_0, S_1, S_2, S_3 are fuzzy similarity relations on X, where X is an initial set of elements, then for one and the same pairs $(x, y) \in X \times X$ a next relationship

$$S_0 \subseteq S_1 \subseteq S_3 \subseteq S_2 \qquad (7)$$

is met. This condition is obvious and follows from conditions (2), (3), (4), (6) and a definition of an inclusion relationship.

From this condition follows, that the usual similarity relation is most general relation for all fuzzy similarity relations on initial set X.

3. The likeness relation and its generalization

Let's consider the likeness relation now. Ruspini defined the likeness relation as follow.

Let S be some fuzzy similarity relation on X. If an ineqality

$$|\mu_S(x, y) - \mu_S(x, z)| \le 1 - \mu_S(y, z), \qquad (8)$$

is met for all $x, y, z \in X$, then S is a likeness relation on X. We will indicate this relation by L symbol.

One in the right component of the inequality (8) is a significance of maximum of membership function of a fuzzy similarity relation, because we can not limit a connection between different elements of X.

We are considering preliminary results as very important properties of fuzzy similarity relations. In particular, properties of projections of fuzzy similarity relations are a basic for generalization of the likeness relation, because clustering relation may be a subnormal relation. In a case of S_0 or S_1 a significance of maximum of membership function of a fuzzy similarity relation is the global projection of feeble similarity relation.

We can generalize the likeness relation from this point of view. This generalization is met for S_0 and S_1 only, if strictly restriction condition (5) is met.

Let S is S_0 or S_1 be some fuzzy similarity relation on X. If an ineqality

$$|\mu_S(x, y) - \mu_S(x, z)| \le \Pr oj(S) - \mu_S(y, z), \qquad (9)$$

is met for all $x, y, z \in X$, then S is generalized likeness relation on X. We will indicate this relation by L_g symbol.

If strictly restriction condition (5) isn't met for S and S is a normal fuzzy similarity relation, then $\Pr oj(S) = 1$. In this case we will consider the likeness relation (8).

Proposition 1. If L_g and L are determined for one and the same $x, y, z \in X$ then a next condition

$$L_g \subseteq L \qquad (10)$$

is met.

Proof. The proof is obvious and follows from (8), (9) and a definition of an inclusion relationship.

Q.E.D.

A next property of L_g is very important for fuzzy clustering.

Proposition 2. If L_g is a generalized likeness relation on X then its addition $D_g = 1 - L_g$ is limited pseudometrics on X.

Proof. The proof follows from a proposition 1 and analogous property of the likeness relation L.

Q.E.D.

4. Summary

We can make some conclusions. Results of investigations are show that feeble similarity relations and strict feeble similarity relations are very interesting and perspective for future investigations, because these relations are soft tools for data representation in fuzzy clustering. We can generalize the likeness relation on a basic of these types of fuzzy similarity relations. However, these results originate some problems, which be required consider for further elaboration of the new methods of fuzzy clustering.

In the first place, a next problem arises. We can normalize fuzzy similarity relation, but this operation can violate the nature of initial data structure sometimes. That's why a problem of normalization of subnormal similarity relation requires special investigation also.

Secondly, the results are show that we must consider is a relationship between generalized likeness relation and its formal representation in fuzzy logic. This problem is require of special investigation, because the problem is very important from theoretical point of view.

References

1. Ruspini E.H. (1982) Recent Development in Fuzzy Clustering. In: Fuzzy Sets and Possibility Theory: Recent Development /Ed. By R.R. Yager.-Pergamon Press, New York.-pp.133-146.
2. Kaufmann A. (1975) Introduction to the Theory of Fuzzy Subsets, Vol. 1, Academic Press, New York.
3. Viattchenin D.A. (1997) Some Remarks To Concept of Fuzzy Similarity Relation for Fuzzy Cluster Analysis. In: Pattern Recognition and Information Processing: Proc. of Fourth International Conference (20-22 May 1997, Minsk, Republic of Belarus). Vol. 1/Ed. by Prof. V.Krasnoproshin et al. -Szczecin, Wydawnictwo Uczelniane Politechniki Szczecinskiej. - pp. 35-38.
4. Viattchenin D.A. (1998) On Projections of Fuzzy Similarity Relations. In: Computer Data Analysis and Modeling: Proc. of the Fifth International Conference (June 8-12, 1998, Minsk). Vol. 2: M-Z/Ed. by Prof. S.A.Aivazyan and Prof. Yu.S.Kharin. - Minsk, BSU. - pp. 150-155.

Adaptive Fuzzy Control Strategies for a Zeppelin

Hubert Roth
Fachhochschule Ravensburg-Weingarten
Institut für Angewandte Forschung
Postfach 1261, D-88241 Weingarten
Germany
Phone: +49-751-501-627
Fax: +49-751-501-654
EMAIL: roth@fbe.fh-weingarten.de

Abstract: The German town Friedrichshafen has an old tradition in building airships. In the moment a follower of the legendary Zeppelin will be build using new technologies. For the control concept this is a challenge because difficult flight manoeuvres are to consider like vertical take off and landing as well as comfortable cruise at different speeds and especially the transition phase between these both manoeuvres. Because of the non exactly known complexity and non-linearity of the airship dynamics good preconditions are given for a fuzzy control concept. The dependency of the dynamics on the strongly variable environmental conditions as for example the airships speed relatively to the surrounding air and the use of different steering components like pivot drives for take-off and landing and the elevation for high speeds requires a high adaptivity of the controller. In this paper the results of a study concerning the usage of a fuzzy concept for control of the pitch axis for all these flight manoeuvres are presented and will be verified by means of realistic simulations.

1. Introduction

Building airships has an old tradition in the German town Friedrichshafen, where in these times a follower of the legendary Zeppelin will be constructed: the Zeppelin NT [1]. An artificial picture is depicted in Figure 1 and some technical details of the airship are listed in Table 1. Typical areas for the use of such an airship at present are [2]:

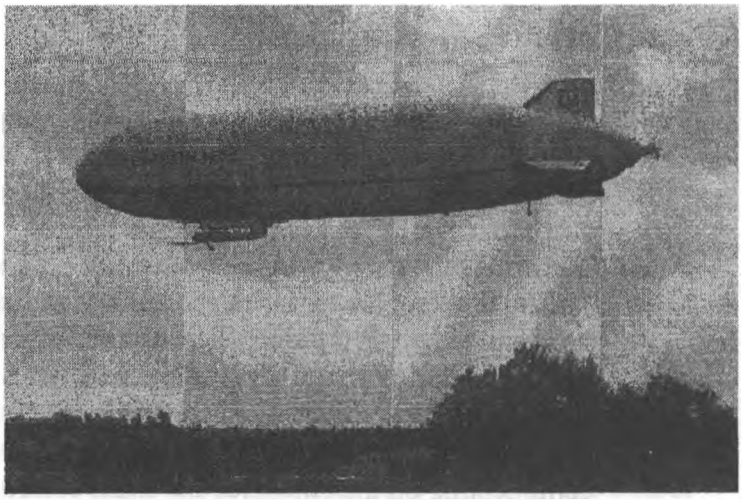


Figure 1: Picture of the Zeppelin NT

- technical applications like environmental monitoring and inspection,
- tourism,
- scientific areas like atmospheric physics and chemistry,
- and last but not least commercial employment so as advertising or TV platform for sport events.

The new airship shall not only be an exact copy of the old Zeppelin, but new technologies shall be incorporated where ever it is possible. To assist the development of a control concept for this airship built with current technology a lot of experience from aeroplane control aspects can be considered. In both cases there are some similar tasks for flight control, such as attitude stabilisation of the yaw-, pitch- and roll- axes, underlying speed control for stability augmentation and complete auto-pilot-functions. No doubt, for the airship the speed range is smaller than for aeroplanes but there are other difficult and completely different flight manoeu-

vres to consider, which complicate a control concept and prevent the simple copying of aeroplane flight control concepts, like:

- vertical take off and landing,
- comfortable cruising at different flight speeds,
- the transition phase between both these flight conditions, and
- finally, the docking manoeuvre at the end of the landing phase with variable strength and wind directions

Table 1: Some preliminary technical dates of the Zeppelin NT

Length	63 m
Diameter	13 m
Weight including maximum payload	5380 kg
Payload	1300 kg
Maximum speed	140 km/h
Maximum flight height	2500 m
Maximum flight time	36 hours

While for cruising at different flight speeds the design of the pitch axis controller with classical methods [3] as well as using a fuzzy concept [4] has been successfully performed, the emphasis of this paper is to extend the design of a fuzzy control concept for the pitch axis to the more complicate manoeuvres "vertical take off and landing" and especially "the transition phase into the horizontal flight".

2. Fuzzy Control Concept

The challenge for a flight control concept of an airship lies in its typical characteristics:

- Related to the size normally it flies with a relatively slow velocity. Therefore the surrounding, partial distributed air conditions have a dominant influence to the flight behaviour which is very badly to model.
- The wings for aerodynamically steering possibilities are relatively small.
- The input variables for influencing the flight behaviour of the airship has continuously to be changed between propeller thrust for vertical take off or landing and the elevator deflection during relatively fast cruise.
- The support structure of the airship is light resulting in high flexibility.
- For balancing the airship the centre of gravity can be well-aimed displaced complicating on the other side the flight control.

But on the other side from these features the following topics result which destine the airship control problem for fuzzy logic:

- the airship is a highly non-linear system,
- a lot of external disturbances which are heavy to model influence the airship,
- even the detailed mathematical / physical model is limited with respect to accuracy
- and finally a control concept has to adapt the variations of different parameters.
- On the other side airship pilots have a detailed flight experience which they can formulate verbally.

While for the control concept during cruising at different speeds due to the aerodynamically dominance the airship is mainly steered by the elevator deflection, the dominant steering devices during vertical take off and landing are three pivot drives oriented perpendicular, one

located at each side of the hull and the third is located at the stern. After turning by 90° they can also be used to increase the cruise speed and improve manoeuvrability. The most challenging control actions are necessary for the transition phase when the pivot drives have to be turned.

The control concept for cruising has the following characteristics: It considers the error signal e (desired pitch-angle minus measured pitch-angle) as well as the angle speed $\dot{\Theta}$ and adapts the fuzzy rules to the true air speed v of the surrounding air. It is supported by the signal "integration of the error". Because for the flight manoeuvre "cruise" the aerodynamically influences dominate the flight behaviour, only the elevator deflection η is used as steering input. All these variables excluding the true airspeed v are devided into 7 fuzzy sets. The rule base consists of some 50 rules.

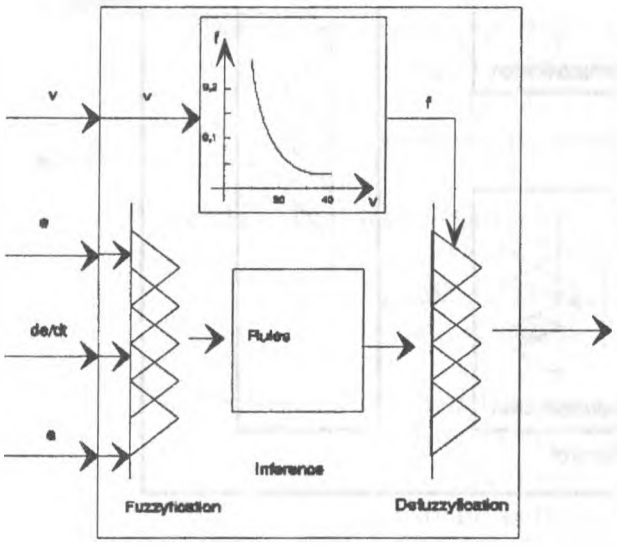


Figure 2: Structure of the Fuzzy-Controller

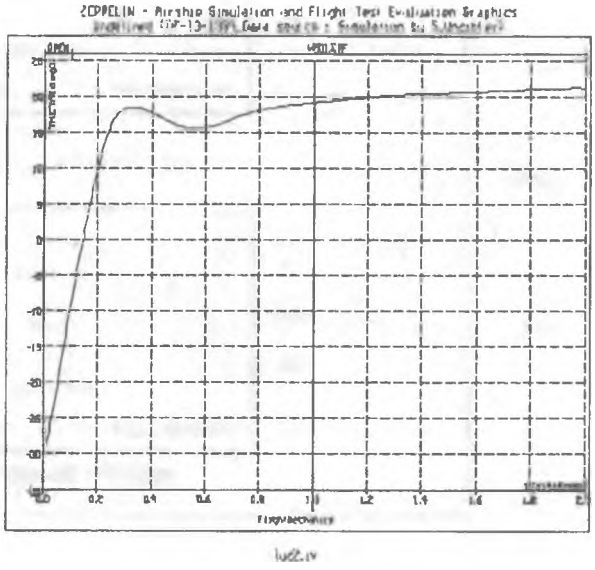


Figure 3: Pitch angle Θ at a climb and reacceleration manoeuvre

The main problem for control is to include the varying speed of the surrounding air relative to the airships speed. In the proposed concept, the adaptivity is provided by varying the width of the fuzzy sets of the controller output "elevator deflection" depending on the true airspeed v. According to Figure 2 a steering factor f is evaluated, where the relationship between f and v is derived by the fact that for a static consideration the overall amplification of the closed loop has to be constant for all airspeeds. Because of the higher effectivity of the elevator deflection during higher speeds the relationship has to be inverse proportional. The width of the output fuzzy set is multiplied by this factor f. The quality of the control is proved by simulation of a reacceleration manoeuvre. For that first the airship is cruising with a pitch angle of -30° for landing. Then the Zeppelin is to be controlled on a pitch angle of 20° simulating reacceleration. The robustness of the controller concept is to be seen in Figure 3, where the pitch angle during this manoeuvre is depicted.

Now, in this paper this cruising controller structure is extended to the vertical take off/landing manoeuvre and the transition phase to cruising. For performing this manoeuvre for the pitch angle control two correcting variables are disposal: In the take-off or landing phase a horizontal speed through the air is missing, so the aerodynamic forces at the flaps have no effect. Therefore the pitch angle is adjusted through the vertical force of the stern drive, turned around 90 degrees. This drive is to be controlled independent of the thrust of the drives on side of the hull. As soon as the airship gathers speed the effect of the elevator flap begins and influence of the vertical force of the stern drive becomes less importance. The controller has to

react appropriately and has to switch between the two correcting variables continuously. For this task a fuzzy controller based on a hyper-inference concept as shown in Figure 4 is designed.

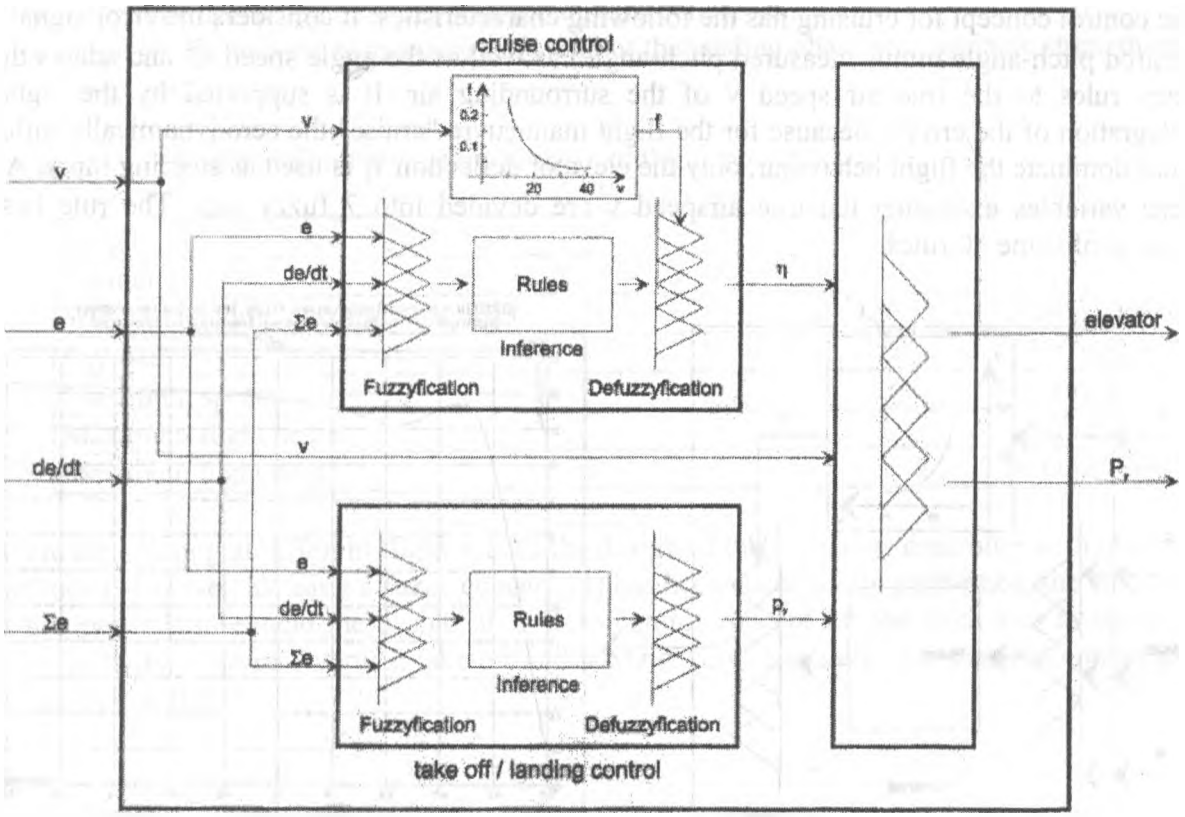


Figure 4: Controller structure for take-off/landing and cruise and the transition phase

4. References

[1] Hagenlocher K. and Mandel M.:
Projekt Zeppelin LZ N07 - Ein Luftschiff neuer Technologie,
Proceedings of Int. Airship-Conference, Stuttgart, 1993

[2] Unzicker S.:
Operational Advantages of Airship over 'Heavier-Than-Air-Craft',
Proceedings of Int. Airship-Conference, Stuttgart, 1993

[3] Adermann, H.-J. and Roth,H.:
Adaptive Fuzzy Control Concept for the Zeppelin New Technology Airship
Proceedings of the Second European Congress on Fuzzy and Intelligent Technologies - EUFIT´94, Aachen, 1994, p.

[4] Roth H. and Adermann H.-J.:
Flight Control Concept of the Zeppelin NT Airship Using Fuzzy Logic
Proceedings of the 10th International Conference on Systems Engineering - ICSE`94, Coventry (UK), 1994

The Architecture of the Neural System for Control of a Mobile Robot

Vladimir Golovko*, Klaus Schilling**, Hubert Roth**, Rauf Sadykhov***,
Pedro Albertos**** and Valentin Dimakov*

*Department of Computers and Mechanics, Brest polytechnic institute,
Moscowskaja 267, 224017 Brest, Belarus
Tel: +375 162 42 10 81, Fax: +375 162 42 21 27
E-mail: cm@brpi.belpak.brest.by

**Steinbeis Transferzentrum ARS, FH Ravensburg-Weingarten, Postfach 1261,
D-88241 Weingarten, Germany
Tel: 0049-751-48542, Fax: 0049-751-48523
E-mail: schi@ars-sun1.ars.fh-weingarten.de

***Belorussian State University of Informatics and Radioelectronics
P. Brovka Str. 6, 220600 Minsk, Belarus
Tel/Fax: +375 172 310 982
E-mail: sadychov@newman.minsk.by

****Department of Systems Engineering and Control
Universidad Politecnica de Valencia
P.O.Box: 22012. E-46071. Spain
Tel: 34-96-3879570 (9570), Fax: 34-96-3879579
E-mail: pedro@aii.upv.es

Abstract

Building autonomous mobile robots has been a primary aim of robotics and artificial intelligence. Artificial neural networks are capable of performing the different aspects of autonomous driving, such as collision-free motions, avoiding obstacles, mapping and planning of path. This paper describes the global architecture of the neural system for autonomous control of a mobile robot. Such neural system has the ability for self-training and self-organizing. The purpose of this paper is to present the key ideas and approaches underlying our research in this area.

1: Introduction

Development of artificial intelligent systems, which are capable to carry out functions of biological beings, is an old dream of humanity. The ability of biological systems to training, self-organizing and adaptation has large advantage as compared with artificial systems. The advantage of computer systems is the high speed of the spreading of signals and the possibility to use large volume of knowledge stored by humanity in various areas. The development of the artificial neural systems, which connect the advantages of computers with the advantages of biological beings, creates the conditions for evolution of artificial sys-

tems to a new qualitative stage. Mostly researches in the area of artificial intelligence are based on the theory of neural networks and are directed at the decision of concrete problems. There is a gradual accumulation of knowledge for creation of universal neural systems. One of the areas, where the creation of "an artificial brain" has large practical and theoretical importance, is the robotics. Mobile robots with capabilities to autonomously reach a target location despite of obstacles are designed for a broad range of applications:

- Transport robots for material transfers in industrial production [1,2]
- Vehicles for planetary surface investigation in the framework of space exploration [3]
- Rovers carrying an equipment for inspection and repair in dangerous environments [4].

This paper is focused on description of an intelligent neural system for the control of a mobile robot. Such system is developed according to the INTAS project (Intelligent Neural System for autonomous control of a mobile robot). Compared to other project activities, the proposed neural system has the ability for self-training and self-organizing and behaves itself as a person during orientation in environment.

2: The Control System architecture

The global architecture of the neural system is represented on Figure 1. It consists of different neural modules, which are combined in an intelligent system. The neural system solves the following tasks:

- Performs data fusion
- Reactive control of the mobile robot while moving in the unknown environment
- The formation of the global route map in the process of the motion in the unknown environment
- The choice of the optimal route and generating

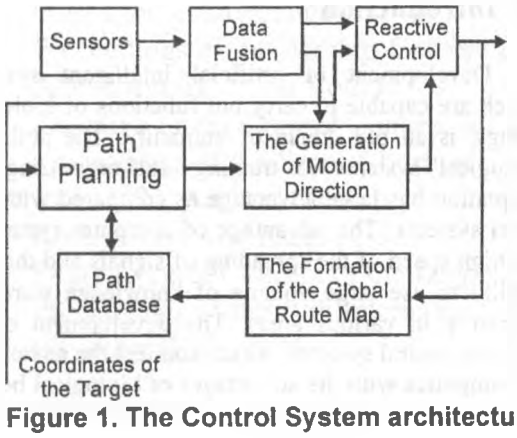


Figure 1. The Control System architecture

The neural system must provide the following demands:

- Robust control in case of inexact information from sensors
- Training with the supervisor
- Self-training and self-organizing
- Capability for real time action

One can see on Figure 1 the information from different sensors is combined by data fusion module. As a result we have the local environment map. The local environment map is used for reactive control and for unpredicted obstacle avoidance, if the working environment is known. Reactive control takes place if the working environment is unknown. In this case, the planning stage has no sense. The inputs to the neural system are the final goal position and the sensor data.

In the process of the robot motion the neural system memorizes the path. For this purpose is used the arrangement of the indicators from start point to target. Each indicator contains direction, which defines how the robot should reach next indicator, a distance between neighbor indicators etc. As a result of robot motion in the unknown environment mapping is performed. Mapping is the process of constructing a model of the environment during motion in the space. As a result of mapping the formation of the global route map and of path database takes place. The path database stores all possible paths and relevant environment data.

Now let's examine the case if the robot motion is performed in the known environment. In this case the path planning module identifies the optimal route for a specific motion action in the actual environment and generates the direction of the motion in the key points (indicators) of the path. For this purpose the path planning module uses a path database to form an optimal solution allowing to reach the target with minimal cost. The neural system performs the reactive control between the key points of the possible route.

Such neural system has ability for self-training and self-organizing. In this case self-training and self-organizing is realized both on the reactive level and on the level of path planning.

3: The hardware platform and sensor fusion

Most of the software has been developed using a mobile robot "WALTER" [5]. The robot, shown on Figure 2, is the LABMATE® mobile robot with a video camera, infrared scanner and ultrasonic transducers. Its maximal velocity is 1000 mm/s. Different

sensors have different perceptual characteristics. As a result of data fusion is turned out the local environment map in the angular interval of 180° and in the review radius of 2.4 meter. The SN288827 Polaroid ultrasonic sensors report distances between 300 mm and 10 m (frequency 45 kHz). As infrared scanner is used the RS2-180 (Leuze electronic). The mobile robot has been designed for indoor environments. An RS-232 radiomodem interface is used to communicate between the SUN Sparc station and the robot microcomputer.

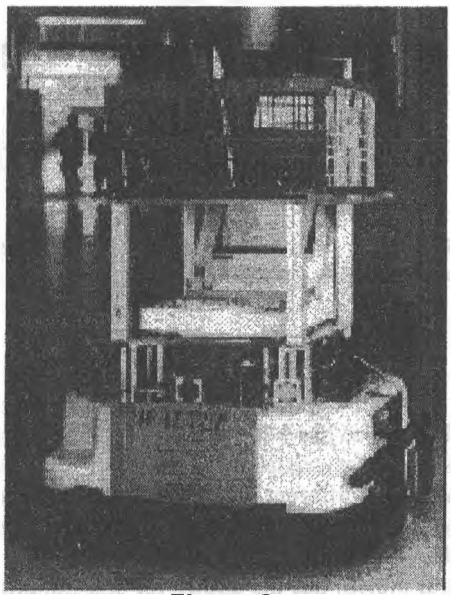


Figure 2.

The analytical approach and neural network are used for data fusion from the ultrasonic sensors and infrared scanner. The analytical approach is as follows: the ultrasonic sensors identify quite exactly the linear distance and the infrared scanner identifies quite exactly the angular distance to an obstacle. The environment map is formed as a result of the simple processing of such an information. The neural network can be used for the sensor error decrease. It consists of 3 layers and is trained on the base of experimentally prepared data. For training is used the backpropagation algorithm with an adaptive step [6]. The outputs of this block is the local environment map which is considered to be input information for the reactive module.

4: The reactive level

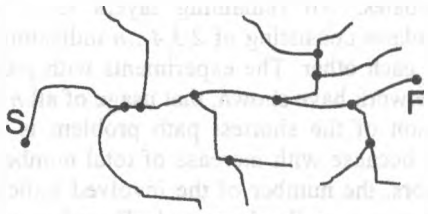
The reactive module consists of various types of neural networks. It provides the robust control of the mobile robot. The reactive module solves the following tasks:

- The selection of optimal interval of motion in environment with obstacles.
- The definition of an optimal direction in the chosen interval of motion

The optimal interval of motion is considered to be the nearest to the target. This interval is characterized to linear (R_L, R_R) and angular (W_L, W_R) distances. The optimal direction is such direction of motion, which ensures minimal angular distance up to the target in the chosen interval of motion. For this purpose the neural networks are used. If one trains a neural network by correct output data in case of inexact input information, it will provide the robust control of the robot. The neural system can itself collect the training data and learn during the interaction of a robot with the environment. As a result the self-organizing of a mobile robot is provided.

5: The formation of the global motion map

It is performed during motion in the unknown environment. The process of map formation is based on memorizing key points of the territory. As key points are used the indicators determining the way of archiving other key points of territory by a robot along the selected path. During examination of unfamiliar territory the transport network (Figure 3) is formed in memory of the planning system. And then the robot uses this information for achieving the target in different parts of territory.



•- an route indicator;
S – start point; F- final point.

Figure 3.

In general the algorithm of territory map formation and planning consists of following steps:

1. If there are two possible motion direction (for example, forward and back), the mobile robot is controlled only by reactive navigation system (robot moves along the corridor in the labyrinth).

2. If there are more then two possible motion directions and there is no any indicator in the current robot position, the planning system creates a new indicator and describes the link to previous indicator as a traversed distance, motion direction to achieve the previous indicator etc. The similar link is created for previous indicator to archive the indicator in current place of the robot from this previous indicator in the future. The navigation system chooses a direction according to the attraction force directed on the target.
3. If there is an indicator on "intersection of roads" in current robot position and there is at least one unknown motion direction, the navigation system tries to examine this direction because there is a possibility to achieve the target using shorter route, but the robot does not known about it yet.
4. If there is an indicator on "intersection of roads" in current robot position and all possible directions are known, the neural networks of best path planning forms an optimal route to the indicator, which is nearest to the target.
5. The indicator is also placed at dead-end situation in a labyrinth.
6. If between two known indicators there is no a free path, the links between them are removed, i.e. territory map is continuously modified.
7. All items are repeated on each step of motion.

6: The generation of the optimal route

It is performed by a neural network solving the shortest path problem (Figure 4). Conceptually it consists of n layers, where n is a number of the indicators memorized by planning system. Here the 1st layer is selecting: it selects a best route from *n-1* routes-candidates. All remaining layers form the routes-candidates consisting of *2,3,4...n* indicators separately from each other. The experiments with proposed neural network have shown, that usage of all n layers for a solution of the shortest path problem is an extreme case, because with increase of total number of the indicators, the number of the involved indicators in the path is essentially decreased. Therefore we used the following equation for determination of number of layers of the neural network:

$$E = \min\left\{\left\lfloor 2 \times \sqrt{n} \right\rfloor + 1, n\right\} \qquad (1)$$

where n is a total number of indicators placed on the territory map.

During we obtained the equal results in comparison with widely reputed Dijkstra's algorithm. However our neural network solved the problem slower because it modelled analogue system. Therefore it is difficult to compare the performance with various pure numerical algorithms.

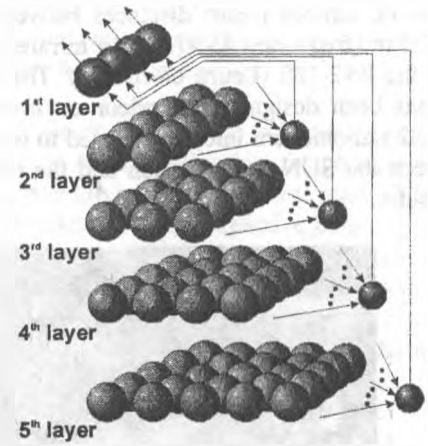


Figure 4. The architecture of the neural network for the problem of 5 cities.

7: Conclusions

An intelligent neural system for the control of a mobile robot has been presented. The ranging system was used for obstacle detection. The neural system consists of different neural networks, which are combined in an intelligent system. This paper describes the global architecture of such a system. The proposed neural system has the ability for self-training and self-organizing. The software will be tested thoroughly using various mobile robots.

8: Acknowledgements

The given work is carried out within the framework of the project INTAS-BELARUS-97-2098 "Intelligent neural system for autonomous control of a mobile robot". The authors thank European Union for the informational and financial support.

References

1. K. Schilling, H. Roth, B. Theobold. Fuzzy Control Strategies for Mobile Robot// Proceedings EUFIT'93, p.887-893.
2. K. Schilling, J. Garbajosa, M. Mellado, R. Mayerhofer. Design of flexible autonomous transport robots for industrial production// Proceedings "IEEE International Symposium on Industrial Electronics", Guirmaraes, 1997, Vol.3,p.791-796.

3. K. Shilling, L. Richter, M. Bernasconi, C. Jungius, C. Garcia-Marirrodriga. Operations and Control of the Mobile Instrument Deployment Device on the Surface of Mars// Control Engineering Practice 5, 1997, p.837-844.
4. T.B. Sheridan. Telerobotics, Automation and Human Supervisory Control// The MIT Press, 1992.
5. K. Schilling, H. Roth. Sensordatefusion mit Fuzzy Logic zur Steuerung mobiler Roboter// In Proceedings of the 2 Neuro-Fuzzy Symposium, Friedrichshafen, Graf-Zeppelin Haus, 1995.
6. V. Golovko, J. Savitsky. Predicting neural net//In Proceedings Intern. Conf. CMNDT-95, Berlin, pp. 348-353.

Reactive Control of a Mobile Robot Based on Neural Networks

Vladimir Golovko
Department of Computers and Mechanics,
Brest polytechnic institute, Moscowskaja 267,
224017 Brest, Republic of Belarus
cm@brpi.belpak.brest.by

Abstract

This paper describes the neural system for reactive control of the mobile robot. The neural system consists of different types of neural networks, which are combined in the intelligent system. The efficient techniques for the training of neural networks are considered. The main problem of the neural network is the robust control of the robot in case of inexact information from sensors. Experimental results are given in the paper.

1. Introduction

One of the most important problems in the design and development of intelligent mobile robots is the robust control, which is the ability of a vehicle to execute collision-free motions in case of inexact information from sensors. Such an approach permits to use inexpensive sensors and to adapt to different environments.

This paper describes the intellectual neural system for reactive control of the mobile robot. The inputs of such a system are the final goal position and the sensor system data. The ultrasonics and infrared scanner is used as sensors. The information from various sensors is combined by data fusion. As a result the local environment map is obtained. The intelligent neural system will process such a map and generate the direction and the velocity of motion. The neural system solves the following tasks:

Sensor data fusion

Building of the local environment map

Obstacle detection and definition of the free interval of motion

Definition of the optimal direction in the chosen interval of motion

The approaches described in this paper can be used for various mobile robots.

2. Architecture of a neural system

The common architecture of a neural system for autonomous control of the robot is shown in fig.1. The system consists of various types of neural networks. In the figure only the main links and blocks of the system are shown. Sensors location is shown in fig.2.

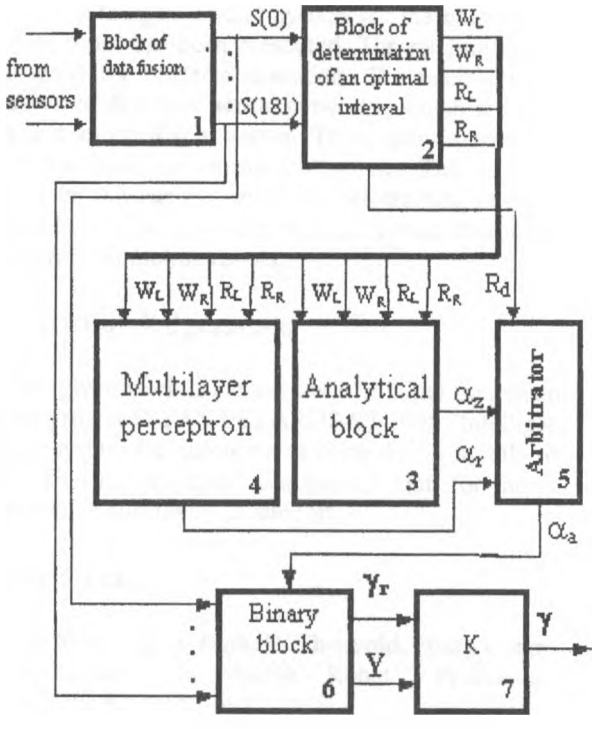


Fig. 1. Neural system for autonomous control of the mobile robot

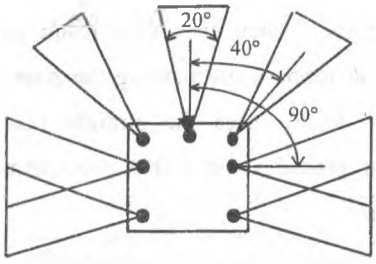


Figure 2. Ultrasonic sensors configuration

The block of data fusion is intended for integrating of different sensors and for the creation of the local environment map. This map is formed in the view radius of 2.4 meter and angular range of 180 grades:

OG = {S(i), -90<=i<=90},

where S(i) is the distance up to the obstacle if the angle between the current direction of the robot and the obstacle is equal i grades. The local environment map is considered to be the input information for the block of the

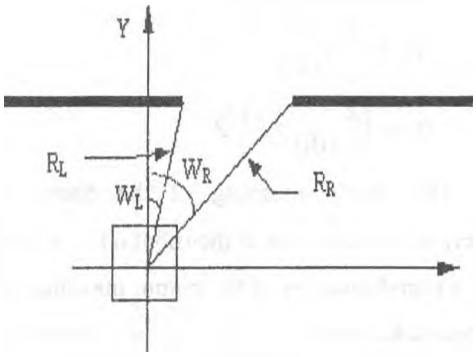


Figure 3.The linear and angular characteristics of the interval

determination of the optimal interval of motion.

Besides this block generates the compressed environment map S(p), $p = \overline{1,36}$, which is used for the control of the binary block.

The block of determination of the interval of motion is intended for the selection of the optimal interval of motion in the environment with obstacles. This interval is considered to be the nearest to the target. The output information of this block corresponds to linear (R_L, R_R) and angular (W_L, W_R) distances of the selected interval of driving (fig. 3).

In case when the free interval of driving is not chosen there is a turning of the robot on 90°, if it is possible and there is the search of the free interval.

Structurally the block 2 consists of 3 layers of neural elements which carry out different functions. This neural network is the dynamic neural network with fixed weights.

The analytical block is meant for the definition of the optimal direction α_z in the selected interval of driving. The optimal direction is characterized by such a direction of driving, which ensures minimal angular distance up to the target. The analytical block controls the motion of the robot on large intervals of driving, if R_d>2d, where R_d is the width of the chosen interval and d is the width of the robot. The architecture of this block consists of different neural layers and processor elements which carry out different functions. The input information of the analytical block is the angle between the current direction and the target, and also angular (W_L,W_R) and linear characteristics (R_L,R_R) of the chosen interval of motion.

The structure and the algorithm of functioning of these blocks was considered in [1-6].

The multilayer perceptron is intended for the orientation of the robot on narrow intervals of driving, where R_d < 2d. It forms the robust direction of motion α_r. The inexact environment map is of great importance for the orientation of the robot on the narrow intervals of motion. If one trains a multilayer perceptron to target output data in case of inexact input information it will provide the robust control of the robot. The input data of this block is the linear (RL,RR) and angular (WL,WR) distances to the obstacles.

The arbiter depending on the situation forms the current direction of the robot:

$$\alpha_a = \begin{cases} \alpha_z, if\ Rd > 2d \\ \alpha_r, otherwise \end{cases} \quad (1)$$

The binary block is intended for the control in the situation, when side distance up to the obstacle $\Delta \le \Delta m$ is too small for the realization of sharp turns. This block transforms the input information to the binary array. The direction, which is formed by the binary block is not higher than 1°. It ensures the avoidance of the contact of the robot with the side obstacles. The commutator depending on the situation forms the final direction of driving of the robot:

$$\gamma = \begin{cases} \alpha_a, if\ Y = 0 \\ \gamma_r, otherwise \end{cases} \quad (2)$$

where Y=1, if $\Delta \le \Delta m$.

Thus depending on the situation the robot can be controlled by the following units:

- Analytical block
- Multilayer perceptron
- Binary block together with the analytical block
- Binary block together with the multilayer perceptron

Such an approach provides stable driving of the robot in various situations. The neural system uses the system of close and long view. The velocity and the step of driving of the robot are normalized depending on the distance up to the obstacle. The break of the robot is performed, if distance up to the target is less than the defined value ε.

3. Multilayer perceptron

The multilayer perceptron is intended for the orientation of the robot on narrow intervals of driving. It provides stable control of the robot in case of inexact information from the environment map. This block forms the robust direction of driving α_r. The architecture of the given block consists of two multilayer networks MLP1 and MLP2 (Fig.4)

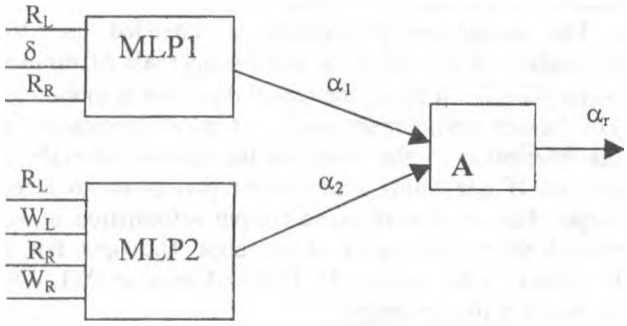


Figure 4. The architecture of the multilayer perceptron

The robust direction of the robot movement is formed by the arbiter:

$$\alpha_r = \begin{cases} \alpha_1, & if \ (R_L \vee R_R) \le g_1 R_t \\ \alpha_2, & if \ (R_L \wedge R_R) > g_1 R_t, \end{cases} \quad (3)$$

where $0 < g_1 \le 1$ – is the constant coefficient; R_t – is the critical threshold of visibility of the accepted system of the view.

According to the expression (3) $\alpha_r = \alpha_1$ if the robot moves in the space between the obstacles and $\alpha_r = \alpha_2$ if the robot moves in the tunnel.

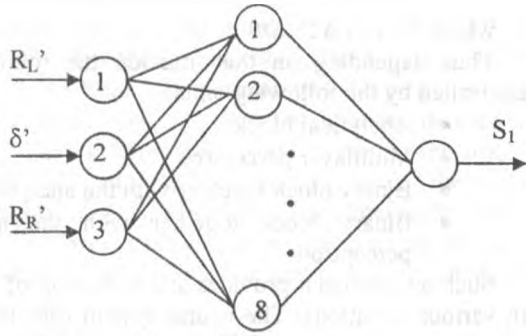


Fig.5 Structure of the block MLP1

Block MLP1 forms the arc of the circumference as the trajectory what secures the exclusiveness of the collision of the robot with the left or right side of the obstacle during the maneuvers. As a result the stable movement of the robot while passing the door ways is realized. Block MLP2 uses the straight line as the trajectory, what provides the stable movement of the robot in tunnels.

Let's examine the structure of these blocks..

Block MLP1 is the 3 layer neural network (fig.5)

It consist of 3 input, 8 hidden and 1 output units. Linear (R_L and R_R) and angular (δ) characteristics of the selected interval of motion are used as the input information. Here $\delta = W_L + W_R$.

Before entering the input of the neural network the data are scaled to the interval [0, 1] according to the following rules:

$$R_L = R_L / 600, \quad (4)$$

$$R_R = R_R / 600 \quad (5)$$

$$\delta = (\delta / 100 + 1) / 2 \quad (6)$$

The output meaning of the neural network characterizes the direction of the robot α1. It is formed by means of transformation of the output meaning S1 of the neural network:

$$\alpha_1 = \text{int}(2S_1 - 1)100. \quad (7)$$

As a result the output meanings of the neural network may alter in the interval [- 100°, 100°]. As the function of activation the sigmoid function is used.

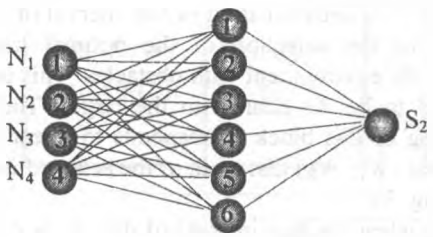


Fig. 6. Structure of the block MLP2

Block MLP2 is also 3 layer neural network (Fig.6). It consists of four input, six hidden and one output units.

Linear (R_L, R_R) and angular (W_L, W_R) distance of the chosen interval of motion are used as the input data. Before entering the input of the neural network the data are scaled as follows:

$$N_1 = R_L / 600, \quad (8)$$

$N_2=R_R/600,$ (9)

$N_3=(W_L/100+1)/2,$ (10)

$N_4=(W_R/100+1)/2.$ (11)

The information on the output of the neural network characterizes a direction of driving of the robot α_2, which is defined as follows:

$$\alpha_2 = \text{int}(2S_2-1)100, \quad (12)$$

where S_2 - the output value of the neural network.

4. The generation of the training set

It is necessary to generate training set for learning of neural networks MLP1 and MLP2. Each learning sample is presented in numeric form and consist of several input and one output meanings. Multilayer perceptron is used for the robot orientation on the narrow intervals of motion, the width of which is less than two meters. The radius of the robot view is 2.4 meters. Therefore it is necessary to generate training set in the following area V:

$$V \in \begin{cases} 1 < R_d \le 2м \\ R_L \le 2.4м \\ R_R \le 2.4м. \end{cases}$$

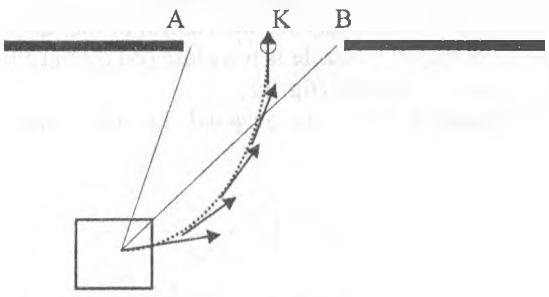


Figure 7 The trajectory of motion

Let's examine the formation of the training set for the block MLP1 .The block MLP1 forms the arc of the circumference as the trajectory of movement. It passes through the center of the robot and through the certain point K in the selected interval of motion (Fig.7) If the coordinates of the point K and the coordinates of the interval of the movement (X_A, Y_A, X_B, Y_B) are known it is possible to define the trajectory of the movement of the robot and the direction of the movement in each point(Fig.7).As a result a lot of learning samples for one position of the robot are formed.

If one performs the rotation of the selected interval of motion [A B] and the rotation of the point K around the center of the robot (Fig.8) it is possible to get different learning samples.

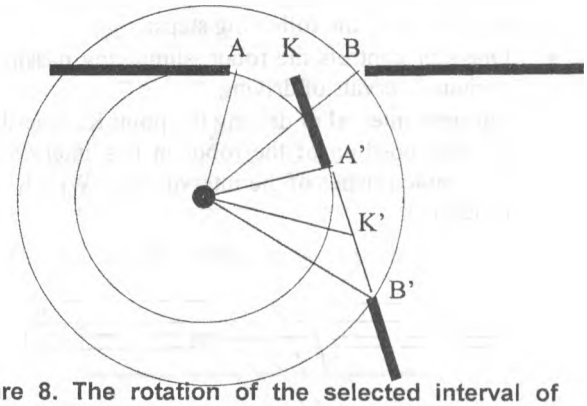


Figure 8. The rotation of the selected interval of motion [AB]

If the position of the robot relatively the interval of motion is changed in the area V and if the operations mentioned above are carried out it is possible to get training set, which consist of different patterns.

The formation of the training set for the block MLP2 is made in the same way. In this case it is necessary in a proper way to select the position of point K as a direction of the robot movement (Fig.9).

If the position of the robot is changed in the area V and if it is rotated with the selected step around the point O (Fig.8) a training set is formed.

In case of the inexact information the real position of the obstacle can differ from the environment map that sees the robot. Such situation is shown in fig 10, where the solid lines represent the position of the obstacle, which the robot sees and dashed - real position of an obstacle.

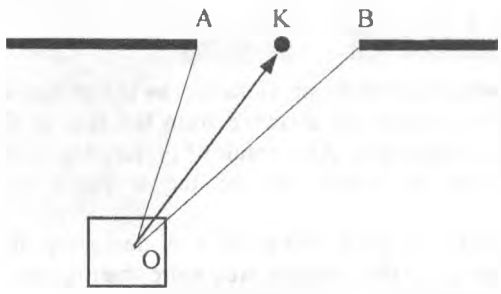


Figure 9. The direction of motion $\overline{OK}$

For the support of robust control of the robot in case of the inexact environment map it is necessary in appropriate way to select a position of point K in the selected interval of driving (fig. 10). If the neural network is trained to

target output data, this can provide stable control of the robot in case of the inexact environment map.

The concept of training of the multilayer perceptron generally consists of the following steps:

- Operator controls the robot, simulating passing of various intervals of driving.
- For each interval of driving the point K, describing the real position of the robot in this interval and the characteristics of the interval (W_L, W_R, R_L, R_R) is defined.

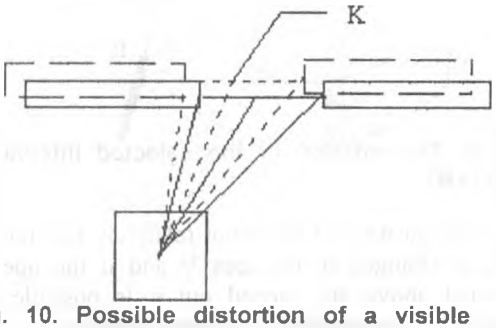


Fig. 10. Possible distortion of a visible interval of driving

- By means of rotation of input and output data in range of 180° learning patterns are formed.
- Training the neural networks MLP1 and MLP2 by means of back propagation algorithm is performed.

The given approach is characterized by the minimum set of the experimental data. It is enough to define only position of a point K and the characteristics of the interval of driving. The computer simulation of the multilayer perceptron was carried out. The size of learning set for the block MLP is 120 patterns. After training the robot successfully passed from various positions through narrow intervals.

5. Binary block

The disadvantage of the previous blocks is that they do not take into account the distance from the side of the robot up to the obstacle. As a result of performing of the maneuvers by the robot can be the collision with obstacles.

To maneuver without collisions it is necessary, that side distance up to the obstacle was more than radius of the circle, circumscribed around of the robot:

$$S > \frac{d}{2}\sqrt{2}, \qquad (13)$$

where d - the width of the robot.

If the condition (13) is not fulfilled, the control of the robot makes the binary block. In this case angle of turn of the robot in any direction is a constant and is equal to one grade.

The binary block is shown in fig. 11.

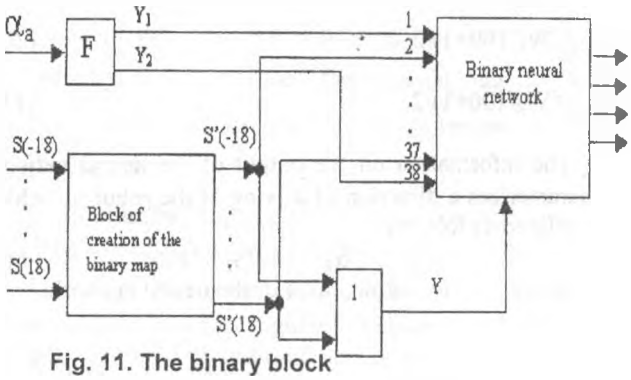


Fig. 11. The binary block

The block F is intended for conversion of the angular direction of driving α_a in binary array. It is necessary for the control of the binary neural network. The block F performs the following functions:

$$Y_1 = \begin{cases} 1, & if\ \alpha_a > 0 \\ 0, & otherwise \end{cases} \qquad (14)$$

$$Y_2 = \begin{cases} 1, & if\ \alpha_a < 0 \\ 0, & otherwise \end{cases} \qquad (15)$$

The block of creation of a binary environment map is intended for generating of the environment map of the given configuration (fig. 12) and formation of the signal Y of activation of the binary neural network.

Such a map is necessary for the control of the robot in situations, when the obstacle is too close (on distance less Δ)to the side of the robot (fig. 12).

The triangular form is selected on the basis of

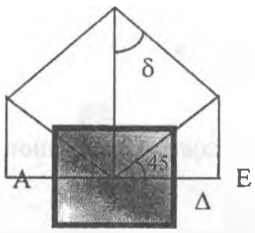


Fig. 12. Configuration of the environment map

providing of smooth maneuvers with the presence of obstacles in front of the robot. As the input information the block of creation of a binary environment map uses the compressed environment map consisting of 36 units. The technology of conversion is, that if the obstacle is in zone ABCDE, the appropriate units S' (p) are installed in single values, otherwise in zero values (fig. 13)

As a result the binary array characterizes the presence of obstacles in the given area. This block consists of one layer of threshold neurons (fig. 14), each of which corresponds to the defined sector of the environment map.

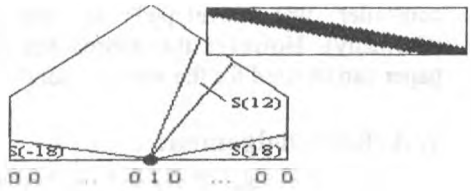


Fig. 13. Example of the binary environment map creation.

The neurons performs the following functions:

$$S'(p) = \begin{cases} 1, & if\ S(p) \le T(p) \\ 0, & otherwise \end{cases} \qquad (16)$$

Here T (p) - the threshold of the given neuron. For creation of the binary array in the given area it is necessary in appropriate way to form threshold values of neurons.

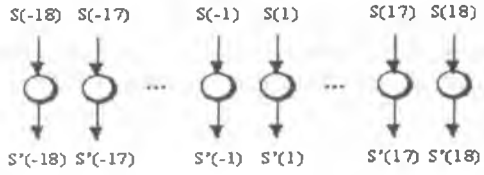


Fig.14. Creation of the binary environment map

The binary neural network is intended for the control of the robot, when the turns on the large values can evolve into collision with the obstacle. In this case Y=1 (fig. 11). Such a network represents the three-layer feed forward neural network (fig. 15).

The sigmoid function is used as the function of activation of units. The commands of the robot are formed by such a neural network(Fug.16).

Thus the turns are performed on 1°, that eliminates

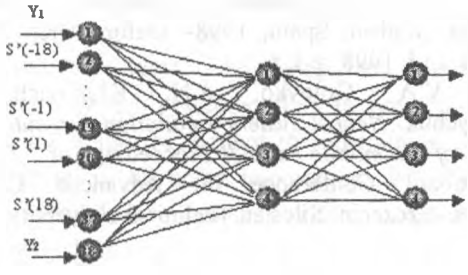


Fig. 15. Binary neural network

collision of the robot with the sides obstacles. For the control of the binary network also the signals Y_1 and Y_2 are used. So, if Y_1=1, that corresponds to α_a>0, the binary network will form the command of turn to the right on the value of 1°. Such an interaction of blocks 3,4 and 5 provides the driving of the robot in the nearest direction to the target. This is especially actual at the existence of the alternate paths of driving in narrow intervals (fig. 17).

The binary network works by the principle of overcoming of the obstacle. The possible variants of its operation are represented in fig. 18

In the second variant the turn takes place until the obstacle is not overcome.

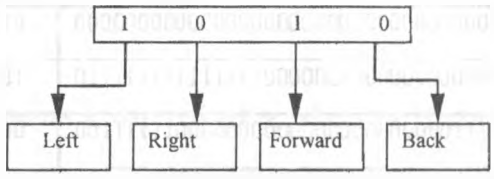


Fig. 16. Control commands of the robot

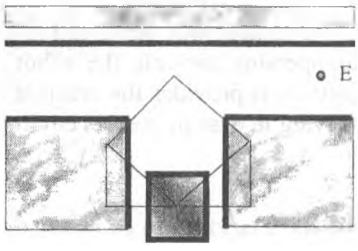


Fig. 17. Driving of the robot in case of the control from the binary block: E - target

Fig. 18 Possible variants of the binary network functioning

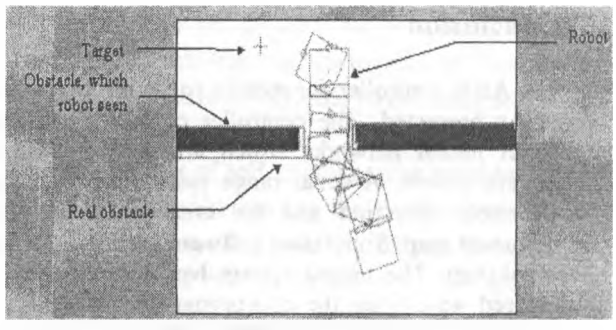


Fig. 19. Robot movement to the target with inexact information from sensor devices

For training of the binary network it is necessary to generate training sets. The generation of learning sampling is characterized by the simplicity and is performed by the logical way. The training sets for some situations are shown in table 1.

For training of the binary network the back propagation algorithm was used. The size of training set is 40 patterns. The regime of modeling can be used

Table 1

Input pattern	Output pattern
01100000000000000000000000000000000001	1010
10000000000000000000000000000000000000	0110
00000000000000000000001111111111111110	1000
00111111000000000000000000000011111100	0010
00000000000000000000000000000000000000	0010
01111111111111111111111111111111111110	0001

together with the logical way for creation of the training set when the operator controls the robot and simulate different situations. It provides the creation of the correct direction of driving in case of inexact environment map.

6. Experiments

For testing of the neural system the software has been developed which allows to simulate the robot motion The training and simulation was performed in case of inexact data from the environment map. For example, the real location of the obstacles differs from the things the robot sees. The tests were carried out for various situations. However learning to target output data will neutralize these inexact data. It provides robust motion of the robot.

Figure 19 shows the experiment where real obstacle differs from the things the robot sees. The tests have shown a good conformity to the theoretical results.

7. Conclusion

The ANN controller for mobile robot reactive control has been presented. The controller consists of different types of neural networks which are combined in the intelligent system. A sonar range system has been used for obstacle detection and for creation of the local environment map. Simulation software aspects have been accomplished. The neural system has been trained in a supervised way, using the backpropagation algorithm. A set of training patterns has been obtained, placing the robot heuristically in different situations. Such a controller was developed for the robot "Walter" (Germany). However the approaches described in this paper can be used for the various mobile robots.

7. Acknowledgement

This work is performed within the project INTAS 97-2028 " Intelligent neural system for autonomous control of a mobile robot". The authors are thankful to the European Community for the financial support of the project.

References

[1] V. Golovko, V. Dimakov and K. Schiling. *Intellectual system for control of mobile robot.* New trends in Artifical Intelegence and Neural Networks: Proc. Int. Conf. (Ankara, Turkey, may, 1997).- Ankara: EMO Scientific Books, 1997, 5p.

[2] V. Golovko and V. Dimakov. *Neural Control System For Mobile Robot.* / Preprints of Intern. Workshop on Intelegent Control INCON'97: (Sofia, Bulgaria, October 13-15, 1997). -Sofia: Technical University, 1997, 5p.

[3] V. Golovko and V. Dimakov. *Intellectual Simulation of Mobile Robot Control System* // Proceedings of the High Performance Computing Symposium, Boston, USA, 1998 – San Diego: The Society for Computer Simulation International, 1998, pp 440-445.

[4] V. Golovko and V. Dimakov. *Intelligent Neural System for Vehicle Control* // Proceedings of the High Performance Computing Symposium, Boston, USA, 1998 – San Diego: The Society for Computer Simulation International, 1998, pp 110 -115.

[5] V. Golovko and V. Dimakov. *Architecture of Neural System for Control of Autonomous Vehicles* // Preprints of the 3rd IFAC Symposium of Intelegent Autonomous Vehicles, Madrid, Spain, 1998– Oxford UK: Elsevier Science Ltd, 1998, v. l, p

[6] V.A. Golovko, A.N. Klimovich, D.Y. Nikolaychuk. *Neural system simulation for autonomous control of the mobile robot.*// Proceedings of The Fifth International Conference on Advanced Computer Systems,–Szczecin: Silesian Technical University.–1998.

Self-training Neural System for Autonomous Control of a Mobile Robot

Vladimir Golovko, Oleg Ignatiuk
Department of Computers and Mechanics,
Brest polytechnic institute, Moscowskaja 267,
224017 Brest, Republic of Belarus
cm@brpi.belpak.brest.by

Key Words: Neural networks, self-organizing, mobile robot.

Abstract

Unsupervised training is of great importance for adaptation to the environment. This adaptation is performed by means of continuos training as a result of interaction between robot and environment. In this case the teacher is the environment. This paper presents the way for the implementation of unsupervised learning and self-organizing. Such approach is offered to be use for a mobile robot.

1. Introduction

One of the most important goals in the design and development of intelligent mobile robots is the ability of a vehicle to adapt to the environment. Life is full of situations, which are impossible to predict. In these cases the ability of a robot to self-training and self-organizing is of great importance. It permits the artificial system to progress without a person (self-progress). It is especially important, whether the robot operates in the aggressive environment or on other planet.

This paper describes a self-organizing system for reactive control of a mobile robot. Such a system must collect a training data into groups itself and learn during the interaction of a robot with the environment. The main principles of self-organizing system are discussed.

2. The general architecture and approach

The general approach to building of the self-training systems consists in the fact, that the initial knowledge of the robot can be filled up and corrected in the process of functioning. It is supposed that the fundamental knowledge of the robot are contained in blocks 1-3 and 6, which are determined by logical way, as it was shown in the previous paper. Then the task is to train multilayer perceptron (block 4) for providing the robust control on the narrow intervals of motion in the process of robot functioning. The scheme of interaction of the robot with environment in the process of self-training is presented on Figure 1. In this case the control is performed from analytical block and binary neural network.

The process of self-training takes place by means of a tries and mistakes on the narrow intervals of motion. If the manoeuvre is carried out successfully the training data for the learning of the multilayer perceptron are formed. If the try is not a success there is a return of the robot to the initial point for several steps back and the repetition of the manoeuvre (Figure 2).

The block of the situation analysis is meant for the reconstruction of the situation on the previous step of the robot (t-1) and the formation of the correcting direction of the movement $\gamma(k_1)$:

$$\gamma(k_1)=\gamma(k)\pm\delta,$$

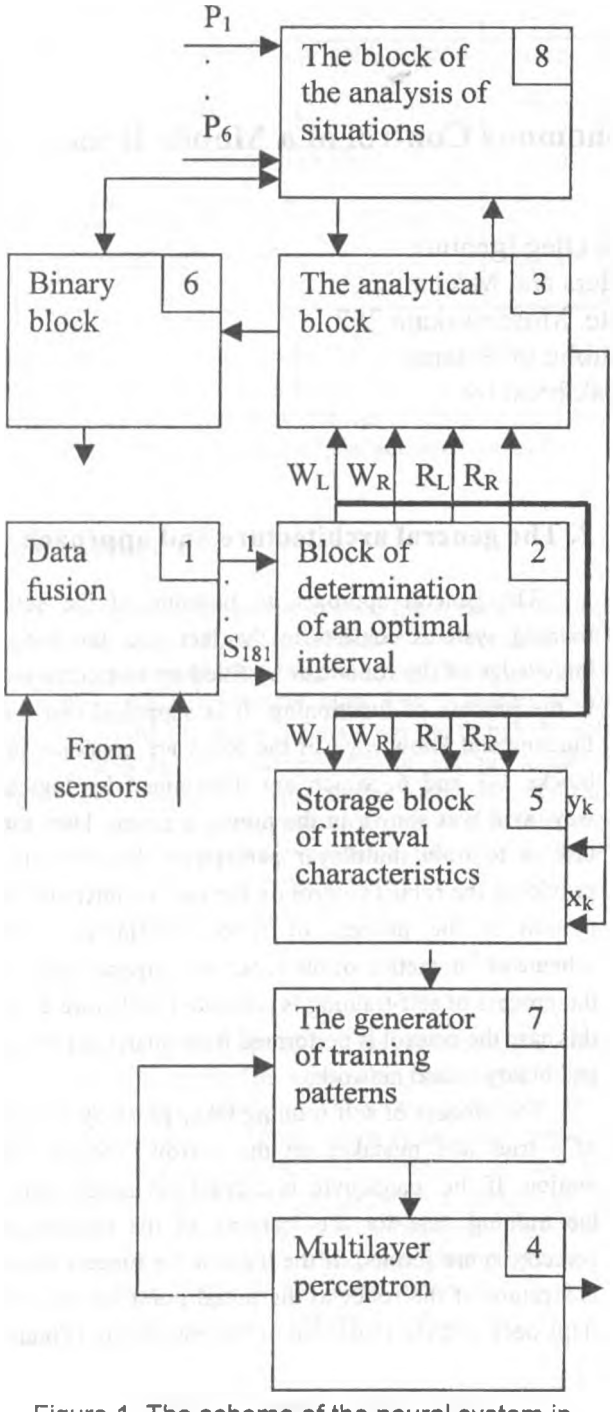


Figure 1. The scheme of the neural system in a self-training mode

where $\gamma(k)$ is the direction of the motion, formed by the analytical block in the given point during the previous try of the manoeuvre; δ is the angel of the correction of the motion direction.

The data from the tactile sensors are used as the input information of the block of situation analysis (Figure 3). By this the input signal of the sensor i equals one (P_i=1), if there has been the contact of the corresponding sensor with the obstacle.

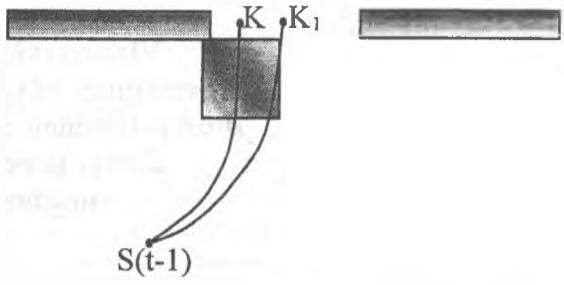


Figure 2. An example of an incorrect trajectory of the robot (point K): K_1 – corrected direction of driving

In the opposite case P_i=0. The correction of the direction of the robot motion is performed by means of logical analysis of the information of the tactile sensors and the previous direction of the movement:

$$(P_1 = 1 \vee P_6 = 1) \rightarrow \delta$$

$$(P_2 = 1 \vee P_3 = 1) \rightarrow -\delta$$

$$((P_4 = 1) \vee (P_5 = 1)) \wedge (y = 1) \rightarrow -\delta$$

$$((P_4 = 1) \vee (P_5 = 1)) \wedge (y = 0) \rightarrow \delta$$

In the expressions given above the signal y is formed as follows:

$$y = \begin{cases} 1, & \text{если } \gamma(k) > 0 \\ 0, & \text{иначе} \end{cases}$$

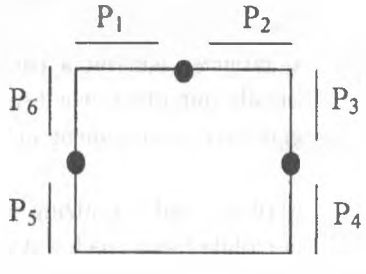


Figure 3. Tactile sensors disposition

Thus the block of the situation analysis forms the positive or negative meaning of the angle of the

correction of the motion direction if there has been a collision with the obstacle. It means that the point K in the selected interval of movement has not been correctly chosen and it is necessary to define the coordinates of the point K_1 according to the new direction $\gamma(k_1)$.

As the binary block operates under the control of the analytical block as a result there is also the correction of output data of the binary neural network. In some situations it is necessary to correct the output meanings of the binary neural network by means of changing of the variant of its functioning. It is carried out by means of the analysis of the information of the tactile sensors and previous output meanings of the binary block. For example, if

$$(P_1 = 1) \wedge K(t-1) = 0110 \rightarrow K(t) = 0100,$$

$$(P_2 = 1) \wedge K(t-1) = 1010 \rightarrow K(t) = 1000,$$

where K(t-1) and K(t) – are accordingly the output meanings of the binary block of the previous and the present stage of functioning. The block of analysis of situations keeps the coordinates of the previous point of movement S(t-1), so as to reconstruct the situation of the previous step. The robot returns in the previous point in case of the collision with the obstacle.

The storage block of interval characteristics contains the coordinates of the present point K and the position of the interval of movement. In case, if the manoeuvre has been a success they enter the generator of the training set.

The manoeuvre is considered to be a success, if the robot reaches the point K in the selected interval of movement without collision with obstacles. The generator by means of rotation defines for blocks MLP_1 and MLP_2 a training patterns, a quantity of which equals approximately to 30, on the basis on coordinates of point K and the characteristics of interval of movement. As a result of modeling of different situations the training set is formed. As the experiments for the stable work of the neural networks have shown necessary volume of the training set equals 120-140. In the process of functioning of the robot it is necessary also the correction of the training set for the binary block. 40 training data are enough for its stable work.

The usage of the binary block for the robot control in the regime of self-training gives an opportunity to decrease the number of mistakes while performing the manoeuvres and consequently to accelerate the process of self-training. By this self-training can take place both for obtaining new knowledge and for correction of the old knowledge. As a result the adaptation of the robot to the environment is provided.

3. Experimental results

For testing of the self-organizing neural system the software has been developed which allows to simulate the robot motion.

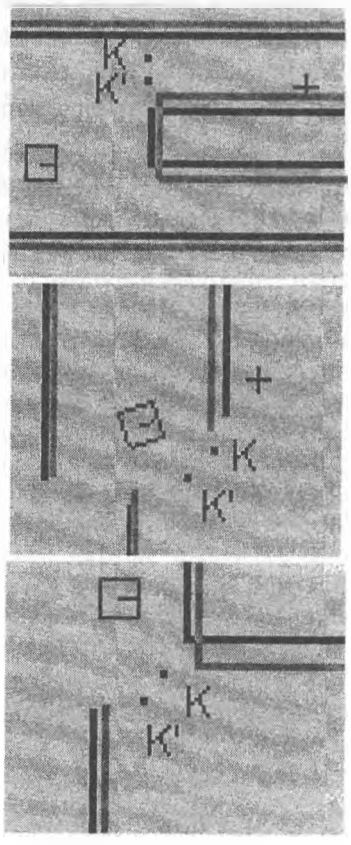


Figure 4. Various kinds of distortions and adjusting direction of the robot; the black color shows real obstacles, and dark-gray - obstacles seen by the robot; a point K is a direction chosen by the robot in the beginning (erroneous), K' – corrected target point of the robot.

The tests were carried out for various situations. By this for the learning and simulation was used inexact data from sensors:

- the linear distances up to an obstacles are differed from real values;
- the angular distances up to an obstacles are differed from real values;
- the sensor errors depend on environment. Therefore the errors of the sensors were changed depending on situation.

The various situation, which were used for simulation are represented on Figure 4. For training multilayer perceptron (block 4) is used the backpropagation algorithm with adaptive step [1]. Such approach permits to reduce a time of training. The experiments have shown, that it takes only several seconds for generating of training set and learning of the multilayer perceptron.

After training the robot motion was a stable for different situations. The self-training permits the robot to reach of a target in case, if the sensor errors are changed during motion. Thus such approach provides the self-organizing of the robot during movement. As a result the robot can adapt to different environment.

4. Conclusion

This paper presents our approaches to autonomous mobile robot navigation. By this during robot motion is performed self-training and self-organization, using the neural networks. The neural networks have been trained in a unsupervised way. During the interaction of the robot with the environment are collected the training data, which are used for training. Such approach permits to adapt the robot to different situations. By this described self-training system works in real time.

5. Acknowledgement

The given work is carried out within the framework of the project INTAS-BELARUS-97-2098 "Intelligent neural system for autonomous control of a mobile robot". The authors thank European Union for the informational and financial support.

References

1. V. Golovko, J. Savitsky. Predicting neural net. In Proceedings Intern. Conf. CMNDT-95, Berlin, pp. 348-353.

Self-Organizing System Forming Strategy of the Global Behavior for Control of an Autonomous Mobile Robot

Valentin Dimakov

Department of Computers and Mechanics,
Brest polytechnic institute, Moscowskaja 267,
224017 Brest, BELARUS
E-mail: VDimakov@mail.ru, cm@brpi.belpak.brest.by,
Fax: +375 162 42 21 27, Phone: +375 162 42 10 81

Abstract

Autonomous mobile robots typically require a preconceived and very detailed navigational model (map) of their intended operating environment. It requires the presence of a priori information. The creating world model of environment is a difficult problem requiring a detailed description of all possible routes of the robot motion. Therefore, it is better to describe a behaviour strategy that robot can create the world model of environment itself during exploration of an unknown territory to achieve efficiently the target from any start position in the future. This strategy assumes full self-organization and self-adaptation to the environment. This paper describes an architecture of such system closely connected with neural network solving the shortest path problem. Such interconnection allows determining the global strategy of the robot behaviour parallel with local strategy formed by reactive navigational system.

1: Introduction

Building navigation systems for mobile robots divided on several fundamental problems: perception and processing of sensor information, reactive control by vehicles and global planning of the behavior strategy.

Building system of the processing of the sensor information assumes the error correction of raw information acting from cheap transducers and combining them in the local environment map. Using the neural network technique allows solving quite effective such problem [1,2].

Together with it, the various neural network models take place at the reactive control by the vehicles. Often the systems of sensor information processing are combined together with reactive control architectures in a single whole [3-5]. Such approach is quite efficiently due to generalization ability of the neural networks.

On other hand, the global strategy of the behavior is oriented on traditional approaches using graph algorithms and on the building Voronoi diagram [6]. Such approaches assume creating detailed world model of the robot safe motion. As algorithm solving shortest path problem the Dijkstra's algorithm and dynamic programming method are used [7,8].

The purpose of this article is to present the key ideas and algorithms underlying research of self-organizing autonomous system to effective control by mobile robots. Existing experience allows defining concepts basing on hybrid technique. In this paper the original approach of building world model of the operation environment is also presented.

2: Principles of the map construction

The planning system consists of two basic subsystems: subsystem of the world model construction and the planning subsystem. The world model is based on indicators placed during exploration of the unfamiliar territory. The navigation system creates indicator as a landmark of the constructed map. As usually, each indicator contains the following information:

- Indicator co-ordinates;
- Serial number of the indicator;
- A set of intervals describing the motion directions formed by the reactive navigation system. Thus, each interval can contain itself parameters, motion direction and the distance to the next indicator (Figure 1). Hence, the robot moves from indicator to indicator until achievement of the target (Figure 2).

Indicator Position and Possible Motion Directions

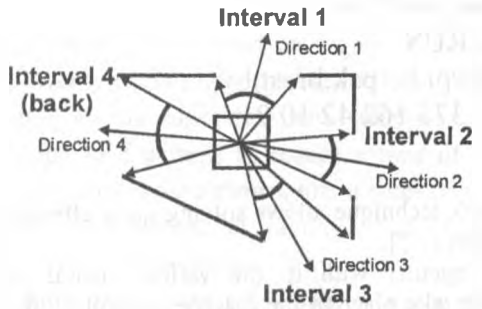
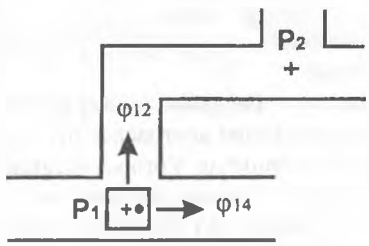


Figure 1.



P_1 – current robot position
P_2 – an indicator
φ_{12} – motion direction to the indicator P_2.
Figure 2.

In general the algorithm of territory map formation and planning consists of the following steps:

1. If there are two possible motion direction (for example, forward and back), the mobile robot is controlled only by the reactive navigation system (robot moves along a corridor in the labyrinth).
2. If there are more then two possible motion directions and there is no any indicator in the current robot position, the planning system creates a new indicator and describes the link to previous indicator as a traversed distance, motion direction to achieve the previous indicator etc. The similar link is created for the previous indicator to achieve the indicator in current place of the robot from this previous indicator in the future. The navigation system chooses a direction according to the attraction force directed on the target.
3. If there is an indicator on "intersection of roads" in current robot position and there is at least one unknown motion direction, the navigation system tries to examine this direction because there is possibility to achieve the target using shorter route, but the robot does not known about it yet.
4. If there is an indicator on "intersection of roads" in current robot position and all possible directions are known, the neural networks, solving the shortest path problem, forms an optimal route to the indicator, which is nearest to the target.
5. The indicator is also placed at the presence of the dead-end situation in the labyrinth.
6. If between two known indicators there is no a free path, the links between them are removed, i.e. territory map is continuously modified.
7. All items are repeated on each step of motion.

There is also a problem of interval definition round the indicator (Figure 3) because the robot does not move exactly from indicator to indicator and it is oriented on an attraction region of the indicator. Thus, the robot can perceive the operation environment state being different from described world model for the given indicator. Hence, the navigation system must "foresee" such situation to avoid continuous modifications of the world model.

The defenition problem of intervals in an attraction region of an indicator

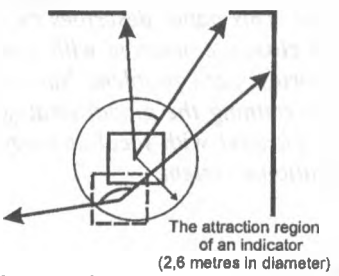


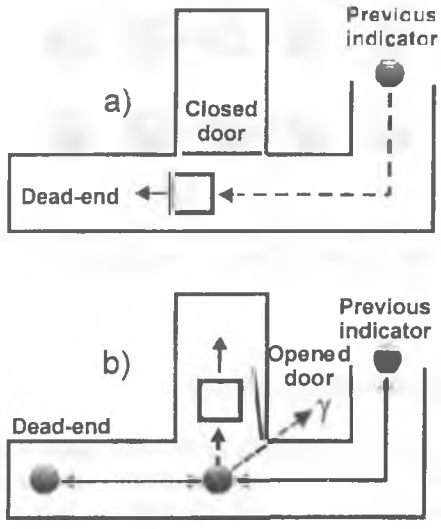
Figure 3. The problem of the interval definition round the indicator.

3: Dead-end situation processing

The main disadvantage of reactive navigation systems is inability to foresee the dead-end situations. Accordingly, the global route planning system must solve this problem. If the robot has already completed world model, the optimal path planning algorithm allows avoiding dead-ends. On other hand, the uncompleted world model does not ensure the successful

achievement of the target. Hence it is necessary to have algorithm allowing solving such problem.
In this case we can define the following actions:

1. In normal work mode, the navigation system forms actions directed on the achievement of the target or of the next indicator in the shortest path.
2. While the reactive system detected dead-end situation, the planning system should form actions to achieve the previous indicator.
3. While the robot comes back from dead-end, it can detect new "intersection of roads" (Figure 4) and therefore it will place a new indicator. In this case there can be several possible motion direction and the robot, choosing one of them, may not achieve the previous indicator. Hence the robot must come back to the new indicator and choose other of possible directions.
4. Coming back from dead-end the navigation system can form new routes if environment state is modified continuously.



a) Before detection of the dead-end situation
b) After detection of the dead-end situation and after placing of a new indicator
γ - direction to come out from dead-end situation

Figure 4.

4: The shortest path formation

The neural network solving effectively the shortest path problem has been developed (Figure 5). Conceptually it consists of n layers, where n is a number of the indicators memorized by planning system. Here the 1st layer is selecting: it selects a best route of $n-1$ routes-candidates. All remaining layers form the routes-candidates consisting of $2,3,4...n$ indicators separately from each other. The experiments with proposed neural network have shown that use of all n layers for a solution of the shortest path problem is an extreme case, because with increase of total number of the indicators, the number of the involved indicators in the path is essentially decreased. Therefore we used the following equation for determination of number of layers of the neural network:

$$E = \min\left\{\left\lfloor 2 \times \sqrt{n} \right\rfloor + 1, n\right\} \qquad (1)$$

where n is a total number of the indicators (key points).

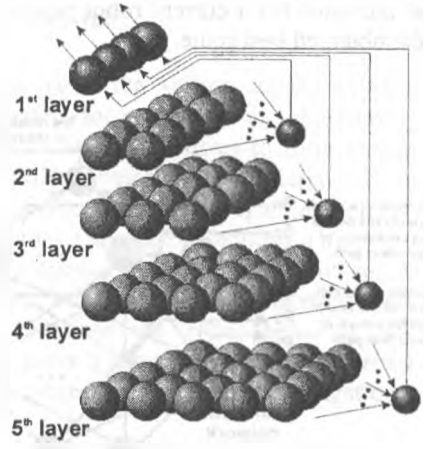


Figure 5. The architecture of the neural network solving the shortest path problem of 5 cities.

One of layers of the network generating a path-candidate is shown in a Figure 6.

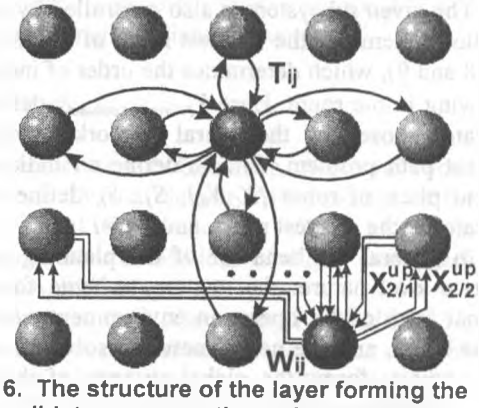
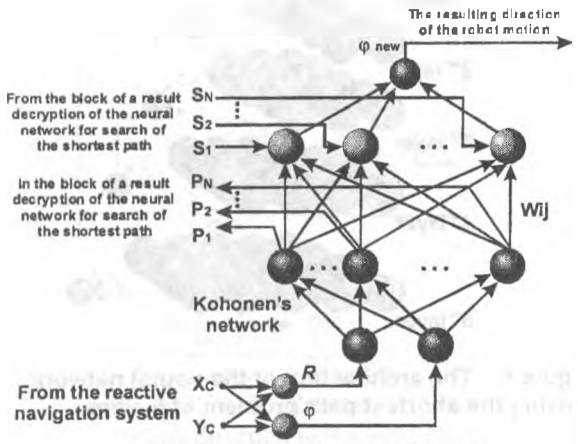


Figure 6. The structure of the layer forming the path-candidate: a connecting schema of neurons by braking (T_{ij}) and exiting connections (W_{ij}).

It is the relaxation network and forms a best route of the robot motion (according to the Figure 6 it forms a route of 4 indicators), where two of them are fixed in the first and the last row of the layer. Here the row number of the layer determines a position of some indicator in the route, and the column number determines a number of the active indicator. At that only one neuron must be active in each row and in each column.

The system shown in a Figure 7 (simplified schema) for memorizing the indicator location on territory was used. It has a Kohonen's network, which is a single-level memory of the storage system of the key points of the territory, and forms also a final motion direction. Here $\{X_C, Y_C\}$- coordinates of current place of the robot, R and φ are polar coordinates of $\{X_C, Y_C\}$, W_{ij} defines a motion direction φ_{ij} (Figure 2) to achieve the next indicator from current robot position according to the obtained best route.



R and φ are polar coordinates

Figure 7.

The given subsystem is also controlled by the decryption scheme of the shortest route of motion (Figures 8 and 9), which determines the order of indicators following in the route. Here $Y_{\Omega<row><column>}$ defines an indicator chosen by the neural network solving the shortest path problem, $P_1 \ldots P_N$ define an indicator in current place of robot $\{X_C, Y_C\}$, $S_1 \ldots S_N$ define a next indicator in the shortest route, and $R_{ij} = 1$.

In general the behavior of the planning system has two-level nature: the memory scheme forms an internal form of the operation environment examined by the robot, and the neural network solving shortest path problem forms the global strategy of the robot motion for achievement of the target in real time.

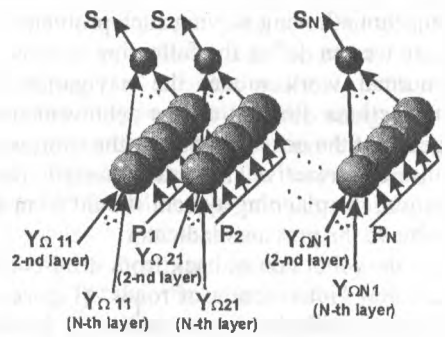


Figure 8. The decryption block of results of the neural network: the connection scheme to the neural network and to the interface scheme.

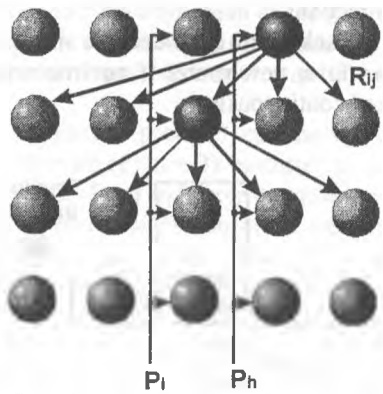
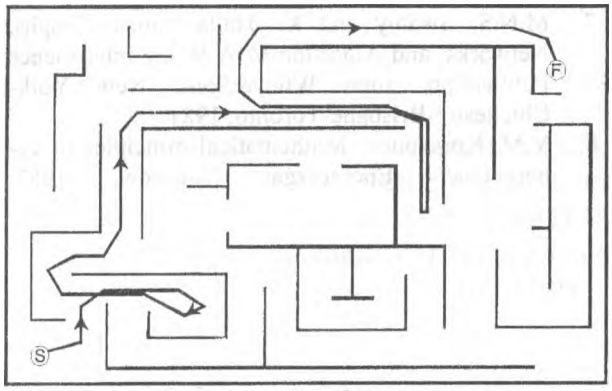


Figure 9. The decryption block of results of the neural network: the scheme of decryption of results.

5: Experimental results

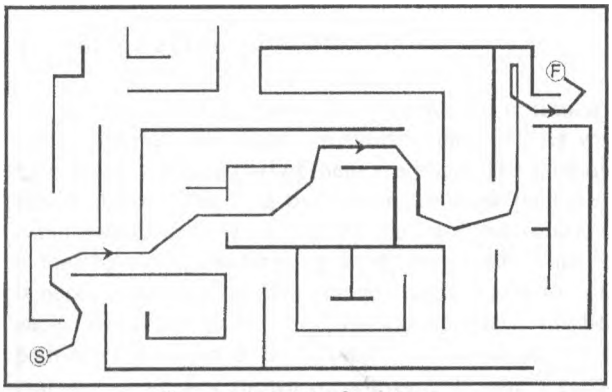
The software combining both the reactive system and self-organizing planning system to test the behavior model was used. As an example the labyrinth image shown in Figures 10-12 was taken. On the first stage the world model formed by the navigation system is clear. At first the robot tries to achieve the final point from start position anyhow (Figure 10). It does not have any a priori information about environment and knows about obstacle arrangement within the limits of sensor device detection (about three meters). During examination of the unfamiliar environment the navigation system places indicators on "intersection of roads". Aftcr first travel the only one possible path detected to achieve the target and it does not cover all possibilities to search an optimal route. During motion in the next time (Figure 11) the robot extends itself knowledge about environment and therefore it can form shorter route like a person in an unfamiliar city.

In the end the navigation system creates a "web" of indicators describing fully the operation environment (Figure 12). If there are changes of the obstacle arrangement, they are emerged and the navigation system modifies itself the environment representation.

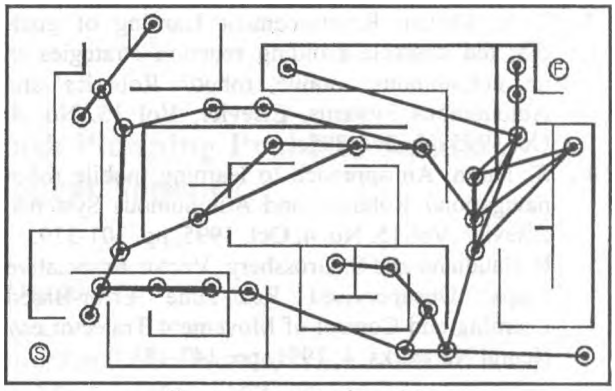


S - start point of motion;
F - final point of motion
Figure 10.

The minimal hardware requirement for real-time simulation for the neural system is PC-computer on the basis of Intel 486DX-100MHz microprocessor. However because of increase of the territory map size the computation delay at the optimal path planning is also increased because of simulation of parallel computations.



S - start point of motion;
F - final point of motion
Figure 11.



S - start point of motion;
F - final point of motion
Figure 12.

6: Conclusions

In this paper the self-organizing optimal route planning system for control of an autonomous mobile robot was considered. It allows to form territory map of an unfamiliar environment, to carry out modification of itself representation of examined environment, to plan optimal route and to control a vehicle in real-time. The proposed approach allows to control by a vehicle in difficult operation environments and at that the system defines the robot behavior without a teacher. At present time the research is carried out to increase robustness of the proposed navigation system.

Acknowledgments

The given work is carried out within the framework of the project INTAS-BELARUS-97-2098 "Intelligent neural system for autonomous control of a mobile robot". The author thanks European Union for the informational and financial support.

References

1. A. Harner and P. Gaudiano. A Neural Model of Attentive Visual Search // Proceedings of the International Conference on Vision, Recognition, Action: Neural Models of Mind and Machine, Boston, MA, May 1997.
2. S. Martens, G. A. Carpenter, P. Gaudiano. Neural sensor fusion for spatial visualization on a mobile robot// SPIE Proceedings, 1998, Vol:3523, pp. 100-111.

3. J. R. Millan. Reinforcement learning of goal-directed obstacle-avoiding reaction strategies in an autonomous mobile robot// Robotics and Autonomous Systems, Elsevier, Vol 15, No. 4, Oct 1995, pp. 275-299.
4. S. Thrun. An approach to learning mobile robot navigation// Robotics and Autonomous Systems, Elsevier, Vol 15, No. 4, Oct. 1995, pp. 301-319.
5. P. Gaudiano and S. Grossberg. Vector Associative Maps: Unsupervised Real-Time Error-Based Learning and Control of Movement Trajectories// Neural Networks, 4, 1991, pp. 147-183.
6. S. Thrun, A. Bücken, W. Burgard, D. Fox, T. Fröhlinghaus, D. Henning, T. Hofmann, M. Krell, T. Schmidt. Map Learning and High-Speed Navigation in RHINO // AI-based Mobile Robots: Case studies of successful robot systems, MIT Press, D. Kortenkamp, R.P. Bonasso, and R.R. Murphy (eds), invited paper.
7. M.N.S. Swamy and K. Thulasiraman. Graphs, Networks, and Algorithms// A Wiley Interscience Publication, John Wiley&Sons, New York-Chichester-Brisbane-Toronto, 1981.
8. Y.M. Korshunov. Mathematical principles of cybernetics// Energoizgat, Moscow, 1987.

The Approach of the Shortest Path Planning Problem Solution on the Basis of a Neural Network

Valentin Dimakov

Department of Computers and Mechanics,
Brest polytechnic institute, Moscowskaja 267,
224017 Brest, BELARUS
E-mail: VDimakov@mail.ru, cm@brpi.belpak.brest.by,
Fax: +375 162 42 21 27, Phone: +375 162 42 10 81

Abstract

This paper describes the neural network solving the shortest path planning problem. It consists of the layer set, where each layer forms a path-candidate for the fixed number of cities. During competition the layer-winner determining the sequence order of cities in the shortest path is selected. In this paper the experimental results obtained by the neural network are described in comparison with some known methods solving the same problem.

1: Introduction

The classical problem of combinatorial optimization is the problem about the shortest path [1]: let we have *n* cities, where two of them are fixed. The goal of this problem is to find the simple shortest path between two these cities. Certainly, the problem solution is the chain of cities forming the shortest path. Thus it is necessary to define their sequence, i.e. a position of each city in the path. The classical approach of the path representation is the arrangement of cities on a matrix $n \times n$ as it is shown on a Figure 1, where each city of *A..E* can have different positions in the shortest path, i.e. *1..4*. Thus this scheme allows uniquely defining the cities-participants and their places in the path. If two cities are fixed then they have the first and the last place in the path correspondingly. Hence we have three-dimensional nature of the problem: at first we have to define the place of each city in the shortest path, and secondly, the number of cities forming the shortest path can be in the range from *2* to *n*.

There are very interesting and efficient approaches solving this problem like famous Dijkstra's algorithm [2] and dynamic programming method [3]. But they are oriented only on solution of the problem on numerical computers. The development of neural network technique is oriented on different technological basis allowing creating efficient systems for different applications.

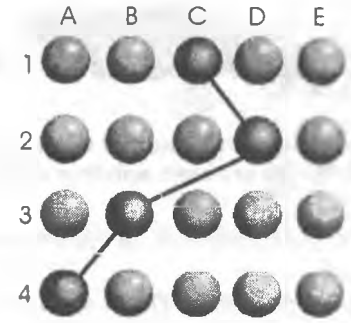


Figure 1. The possible scheme of the arrangement of cities in the path between cities "C" and "A".

There is also a possibility to solve this problem using the neural network technique like the Hopfield's neural network [4]. The solution using the neural network like that is enough difficult and it is defined by a weight matrix as follows [5]:

$$T_{AiBj} = -\alpha\delta_{Aa}\delta_{Ba}\delta_{i1}\delta_{j1} - 2\beta\delta_{Ab}\delta_{Bb} + \beta\delta_{Ab}\delta_{ij} -$$
$$-\gamma\delta_{ij}(1-\delta_{AB}) - \eta\delta_{AB}(1-\delta_{ij}) - 2\theta\delta_{ij}(1-\delta_{Ab})*$$
$$*(1-\delta_{Bb})(1-\delta_{AB}) - \theta\delta_{ij}\delta_{AB}(1-\delta_{Ab}) - [2(1-\delta_{Aa})* \quad (1)$$
$$*(1-\delta_{AB}) + \delta_{AB}]\theta\delta_{ij}(1-\delta_{Ba})(1-\delta_{i1}) - 2\theta\delta_{j1+1}*$$
$$*(1-\delta_{Ab})(1-\delta_{Ba}) - \chi(1-\delta_{AB})C_{AB}(\delta_{j1+1} + \delta_{j1-1}),$$
$$U_{Ai} = -\alpha\delta_{Aa}\delta_{i1} - \beta\delta_{Aa}$$

where

$$\delta_{ij} = \begin{cases} 1, if\ i = j \\ 0, otherwise \end{cases} \quad (2)$$

The difficult here is to define constant parameters $\alpha, \beta, \gamma, \eta, \theta$ and χ. In this paper the simpler neural network approach solving efficient this problem is considered. On basis of the problem condition the general architecture of the proposed neural network is defined (Figure 2).

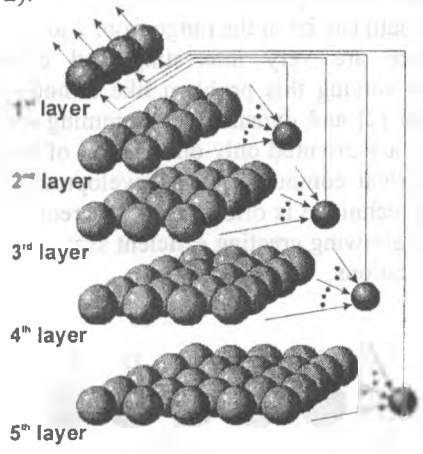


Figure 2. The architecture of the neural network for the problem solution of 5 cities.

2: The neural network architecture

Generally the neural network consists of n layers, where n is a total number of cities. The first layer selects the best variant of paths-candidates, which are offered by the remaining layers. The second layer forms the path-candidate consisting of two cities; the third one forms the path-candidate consisting of three cities etc.; and, at last, the n^{th} layer forms the path-candidate consisting of n cities, where two of them are fixed. At least one path-candidate of $n-1$ offered must exist for successful solution.

The structure of the first layer is shown on a Figure 3.

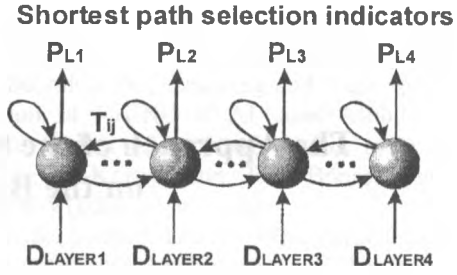


Figure 3. The structure of the first layer.

It is the single-layer relaxation competing neural network determining the neuron-winner having the output value equal to "1", and the all-remaining neurons have output value equal to "0". As input value for each neuron the value describing length of each offered path is used. The braking and self-exciting connections T_{ij} interconnect the neurons among themselves. Their weight values are follows:

$$T_{ij} = \begin{cases} 1 & , if\ i = j \\ -1/(m+1), & otherwise \end{cases} \quad (3)$$

where m is a network dimension. The iterative process of such neural network is described as:

$$Pl_i(0) = D_{LAYER\,i} \quad (4)$$

$$Pl_i(t+1) = fn\left(\sum_{j=1}^{m} T_{ij} * Pl_j(t)\right), \quad (5)$$

where

$$fn(x) = \begin{cases} 0 & , if\ x \le 0 \\ 1 & , if\ x \ge 1 \\ x & , otherwise \end{cases} \quad (6)$$

Here D_{LAYERi} is a parameter describing a length of path formed by other layer.

During an iterative process one of neurons becomes the winner determining the best path.

The important components of the neural network are the layers forming the paths-candidates consisting of different number of cities, i.e. $2,3,..,n$ (Figures 4 and 7). Hence the neural network has a pyramidal form.

The schema of such layer is similar to the scheme of the city arrangement in the path (Figure 1). Here each neuron has a set of braking T_{ij} and exiting W_{ij} connections. The braking connections determine state when only one neuron can remain active in the each column and in the each row. Their weight factors are determined as:

$$T_{1ik} = \begin{cases} 0 & ,if \ \ i = k \\ -2/(n+1), & otherwise \end{cases} \quad (7)$$

$$T_{2ij} = \begin{cases} 0 & ,if \ \ i = j \\ -2/(p+1), & otherwise \end{cases} \quad (8)$$

where n is a number of neurons in the row, and p is a number of neurons in the column.

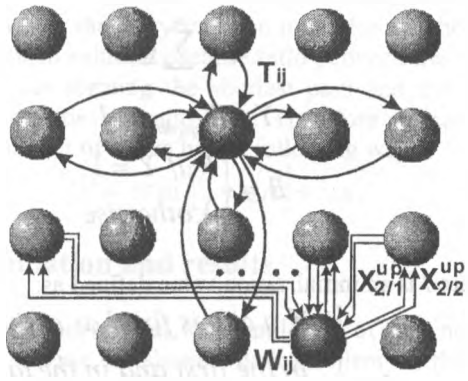


Figure 4. The structure of the layer forming the path-candidate: a connecting schema of neurons by the braking and by the exiting connections.

The formation of the shortest path in each layer is carried out according to the weight factors of exiting connections W_{ij} defined a priori:

$$W_{1ij/1} = W_{2ij/1} = \begin{cases} 0 & ,if \ \ i = j \\ 1/d_{ij}, & otherwise \end{cases} \quad (9)$$

$$W_{1ij/2} = W_{2ij/2} = 1 \quad (10)$$

where d_{ij} is a distance between i^{th} and j^{th} cities, W_{1ij} is a connection weight factor between a neuron of k^{th} row and a neuron of $(k\text{-}1)^{th}$ row, W_{2ij} is a connection weight factor between a neuron of k^{th} row and a neuron of $(k+1)^{th}$ row correspondingly. The weight factors W_{1ij} and W_{2ij} are the same for all rows of the layer. Thus the first and the last row of the layer does not have the exiting connections between themselves.

Each layer has two sorts of neurons: intermediate neurons are placed on intermediate rows of the layer (Figure 5) and final neurons are placed on the first and on the last row of the layer (Figure 6). Both sorts of neurons have a complex structure allowing forming the path-candidate in appropriate conditions for each layer.

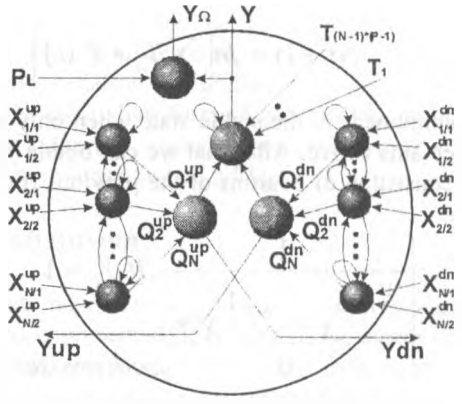


Figure 5. The scheme of the intermediate neural element.

The intermediate neuron (Figure 5) has two low-connected subsystems. Each of them obtains the information through exiting connections from neurons of the previous row and propagates it on the next row. Thus, the information propagates through each neuron both "top-down" and "down-top". The intermediate neuron contains two competing neural networks working similar to the neural network of the first layer (Figure 3). Their input values X_i are formed by product of output values of neurons of the previous row $Y_{up(dn)}$ on appropriate weight factor $W_{1(2)ij}$.

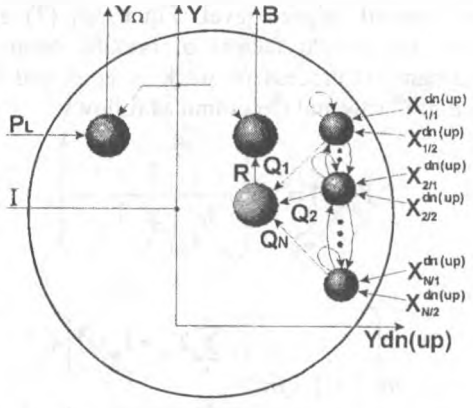


Figure 6. The scheme of the final neural element.

Thus, the dynamical process of the work of the competing neural network of a left part of the neuron (Figure 5) can be described as follows:

$$Y_i(0) = \frac{1}{X^{up}_{i/2}/X^{up}_{i/1} + 1/X^{up}_{i/2}} \quad (11)$$

$$Y_i(t+1) = fn\left(\sum_j T_{ij} * Y_j(t)\right) \quad (12)$$

It is continued till the stable state when only one neuron remains active. After that we can obtain weight of path consisting of neurons of the previous rows of the layer:

$$Q_i^{up} = \begin{cases} \dfrac{1}{X_{i/2}^{up} / X_{i/1}^{up} + 1 / X_{i/2}^{up}}, & if\ Y_i = 1 \\ 0, & otherwise \end{cases} \quad (13)$$

The dynamical process for the right part of the neuron is the same.

The information for the subsequent rows of the layer is formed as follows:

$$Y_{up} = \sum_i^n Q_i^{dn} \quad (14)$$

$$Y_{dn} = \sum_i^n Q_i^{up} \quad (15)$$

After definition of activity of each neuron in the layer it is necessary to define in fact the path-candidate for the layer. For this purpose the rules are defined when only one neuron remains active in each row and in each column. Such state is defined by the relaxation neural network higher level. Equations (7) and (8) describe the weight factors of braking connections. The dynamical process of work is described for the neuron of i^{th} row and j^{th} column as follows:

$$Y_{ij}(0) = fn\left(\frac{1}{1/Y_{up} + 1/Y_{dn}}\right) \quad (16)$$

$$Y_{ij}(t+1) = fn\left(Y_{ij}(t) + fn2\left(\left(\sum_k^n T_{1ik} * Y_{kj}(t)\right) + \left(\sum_s^p T_{2sj} * Y_{is}(t)\right), Y_{ij}(t)\right)\right) \quad (17)$$

where

$$fn2(x,y) = \begin{cases} x, if\ y \geq 0.5 \\ 0, otherwise \end{cases} \quad (18)$$

The output value Y_Ω of each neuron is a part of the final solution of the neural network. It specifies activity of a neuron of the selected layer and it is determined as follows:

$$Y_\Omega = \begin{cases} Y, if\ P_L = 1 \\ 0, otherwise \end{cases} \quad (19)$$

where P_L is signal acting from the first layer. It determines *the layer-winner*.

In comparison with the intermediate neuron the finite neuron (Figure 6) contains only one relaxation network and an element forming a value B describing the length of the generated path. The work of the finite neural element can be described as:

$$R = \sum_i^n Q_i \quad (20)$$

$$Y = Y_{up(dn)} = I \quad (21)$$

$$B = \begin{cases} R, if\ Y = 1 \\ 0, othewise \end{cases} \quad (22)$$

Here I is an initialization value defined as:

$$I = \begin{cases} 1, if\ a\ city\ is\ fixed\ as\ active \\ \quad in\ the\ first\ and\ in\ the\ last \\ \quad row\ of\ the\ layer \\ 0, otherwise \end{cases} \quad (23)$$

At the end of the process forming the path in each layer the special elements form the value D_{LAYER} characterizing the length of the generated path (Figure 7).

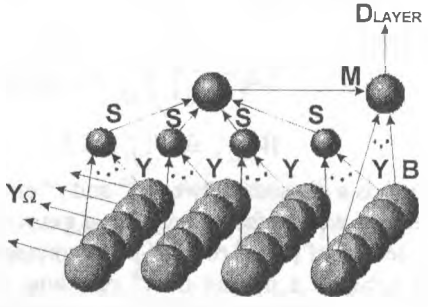


Figure 7. The schema of formation of the D_{LAYER} value.

It is determined as follows:

$$S_k = \sum_i Y_{ki} \quad (24)$$

$$M = \prod_k S_i \quad (25)$$

$$D_{LAYER} = \begin{cases} \sum_{i}^{n} B_i, if\ M = 1 \\ 0 \quad , otherwise \end{cases} \qquad (26)$$

where k is a row index in the layer and M indicates the successful formation of the path-candidate in the layer.

The total number of neural elements of the neural network is:

$$T = \left(\sum_{1}^{n} i * n \right) - 1 = \frac{n^3}{2} + \frac{n^2}{2} - 1 \qquad (27)$$

However the n layers is too much for the shortest path problem solution because ratio between the number of cities forming the shortest path and the total number of cities is quite little. Therefore we can reduce a number of layers by the following way:

$$E = \min\left\{ \left\lfloor 2 \times \sqrt{n} \right\rfloor + 1, n \right\} \qquad (28)$$

3: Simulation and results

It was carried out the simulation of this neural network solving the shortest path problem on the sequential personal computer with processor Intel Pentium II-350 under control of the Windows NT v4.0 operating system. The modeling algorithm has been optimized for sequential computer to reduce the computation time. The model of the neural network was tested on solution of 4 sorts of tasks: 5, 15, 50 and 100 cities. The solution average time presented in seconds is shown in the Table 1 in comparison with other two famous methods for each sort of tasks.

Table 1.

Simulation method	*Size of the problem (a number of cities)*			
	5	**15**	**50**	**100**
Dynamic programming method	0	0	0	0
Dijkstra's algorithm	0	0	0	0
The neural network	0	0,02	1,58	22,03

As obvious from the Table 1 the model of the neural network works slower than pure numerical algorithms because it simulates analogue processes. Nevertheless it has formed good solutions for all 4 sorts of tasks (for example, see results in the Table 2 for the map presented on a Figure 8).

4: Conclusion

The neural network solving the shortest path problem is presented. It is used in the system forming the global route map during the motion in the unknown environment. The neural system has difficult pyramidal structure allowing forming simple shortest path to control efficiently by a mobile robot motion. This paper describes the architecture of such neural network. The experimental results have shown effectiveness of the proposed model.

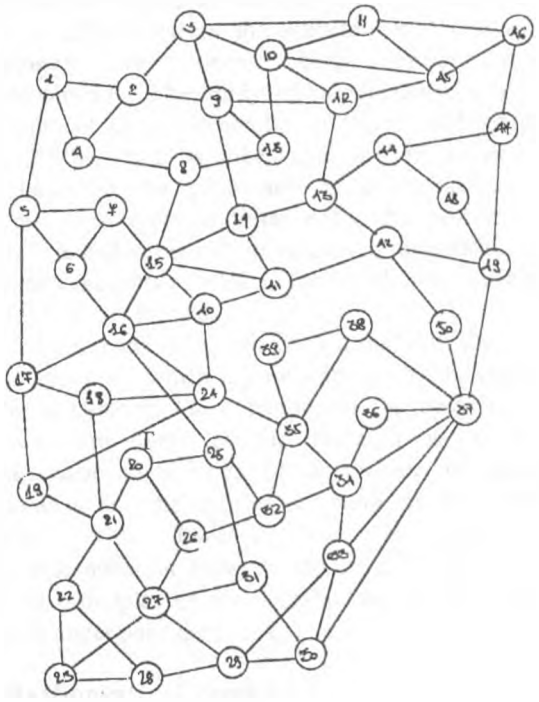


Figure 8. An example of the map of 50 cities.

Table 2.

Task num.	*Dynamic programming method*		*Dijkstra's algorithm*		*Neural network*	
	Solution	*Path length*	*Solution*	*Path length*	*Solution*	*Path length*
1	23-28-29-33-37-49-47-46	213	**23-27-26-32-34-37-49-47-46**	**209**	**23-27-26-32-34-37-49-47-46**	**209**
2	30-31-25-16-15-8-4	158	30-31-25-16-15-8-4	158	30-31-25-16-15-8-4	158
3	35-24-40-41-14-9-3	125	35-24-40-41-14-9-3	125	35-24-40-41-14-9-3	125
4	23-28-29-33-37-49-48-44	195	**23-22-21-19-17-16-15-14-43-44**	**190**	**23-22-21-19-17-16-15-14-43-44**	**190**
5	45-12-43-42-50-37-30	179	45-12-43-42-50-37-30	179	45-12-43-42-50-37-30	179
6	17-16-40-41-42-49	119	17-16-40-41-42-49	119	17-16-40-41-42-49	119
7	2-9-14-43-42-50-37	132	2-9-14-43-42-50-37	132	2-9-14-43-42-50-37	132
8	11-10-13-8-15-16-25-31	177	11-10-13-8-15-16-25-31	177	11-10-13-8-15-16-25-31	177
9	28-29-33-37	102	28-29-33-37	102	28-29-33-37	102

5: Acknowledgements

The given work is carried out within the framework of the project INTAS-BELARUS-97-2098 "Intelligent neural system for autonomous control of a mobile robot". The author thanks European Union for the informational and financial support.

References

1. Michael R. Garey and David S. Johnson. Computers and intractability// San Francisco, Bell Telephone Laboratories, Inc., 1979.
2. M.N.S. Swamy and K. Thulasiraman. Graphs, Networks, and Algorithms// A Wiley Interscience Publication, John Wiley&Sons, New York-Chichester-Brisbane-Toronto, 1981.
3. Y.M. Korshunov. Mathematical principles of cybernetics// Energoizgat, Moscow, 1987.
4. J.J. Hopfield and D.W. Tank. Collective computation with continuos variable// Disordered systems and Biological organization, Springer-Verlag, 1986, pp. 155-170.
5. I.I. Melamed. Neural network and combinatorial optimization// In Proceedings of the 16th IFIP Conf. Syst. Model Optim., Compieqne, France, 1993, pp. 537-542.

The Regularities of the Development of the Nonlinear Dynamics of the Control Systems with Pulse-Width Modulation

Yuri V. Kolokolov, Stanislav L. Koschinsky
Radioelectronics Department, State Technical University of Orel,
29, Naygorskoye Shosse, 302020 Orel, Russia,
tel: 7 0862 421661, fax: 7 0862 416684, e-mail: kolo@valley.ru

Abstract. The paper reports on the dynamics of the control systems with pulse-width modulation of the second kind. The bifurcations of the stationary motions are studied. The regularities of occurrence of chaotic processes are revealed.

Key words. Pulse control systems, nonlinear dynamics, bifurcation, chaos.

1. Introduction

The control systems with pulse-width modulation (PWM) belong to the class of essentially nonlinear dynamic systems of variable structure. The use of the control systems with pulse-width modulation in many directions of human practice such as transport, mining industry, metallurgical and coal industry, etc. are related with the solution of the problems of qualitative transformation of great quantities of energy (from one or even less then one kW to hundreds thousands kW). In this case the compulsory requirements to the dynamics of the process of transformation of energy are as follows:

- ensuring of efficiency;
- safeguarding of security, namely, the forecast and elimination of possible emergency (catastrophic) situations;
- ensuring of compatibility (including electromagnetic) with environment. But the efficiency of the process of transformation is in contradiction with the requirements of security and compatibility.

The realizations of these requirements complicated by the fact that many control systems with PWM according to the request for the proposal run under conditions of variation in their own parameters or in the parameters of conjugate systems over a wide range.

It is known [1,2] that for different combinations of parameters the dynamics of the control systems with PWM may be regular or chaotic. In this case stationary motions, different from the predetermined one while designing, may occur both "mildly" and catastrophically. The study of the regularities of the development of the dynamics of the control systems with PWM, that is, the analysis of bifurcation boundaries in the parameter space of the control systems is of great importance for the problems of parametric and structural optimization, identification of stationary motions as well as for the choice of optimal control in such systems.

This research has been carried out with the scope of the approach, which is being formed by the authors, to the analysis of the dynamics of the pulse control systems. The aims of the research is to show characteristic regularities of occurrence of chaotic phenomena for the typical structure of the control systems with PWM-2 and to find the intercommunication between the revealed regularities and the principles of the constructions of the control systems under considerations.

2. Mathematical model

The mathematical model of the standard structure of control system with PWM [1,2] has the form

$$\frac{dX}{dt} = G(X) + B(K_F), \qquad (1)$$

where X is the vector of state variables of the control system; the vector function G(X) and the vector $B(K_F)$ describe the elements of the control system which are invariant in time; K_F is the pulse function, which determines the state of the key element of the control

system in accordance with the sign of the function of commutation

$$\xi(t, X(t)) = \alpha \cdot (U_c - S' \cdot X(t)) - U_0 \left(\frac{t}{a} - E_1\left(\frac{t}{a}\right) \right), \quad (2)$$

accordance to the algorithm

$$K_F = \begin{cases} 1, & \xi(t, X(t)) \geq 0, \\ 0, & \xi(t, X(t)) < 0. \end{cases} \quad (3)$$

In equations (2) and (3) α is the coefficient of amplification of the error signal; U_C is the set point; S' is the vector line, which establishes the agreement between the state variables of the control system and the value at the input of the pulse modulator; U_0 is the amplitude of saw tooth voltage; a is the timing period of the pulse-width modulation; $E_1(.)$ is the function whose result represent a whole part of the argument.

The moment of the change of the structure of the control system t_k (the commutation time) during the period of the PWM $(k-1)\cdot a < t < k\, a$ is determined in accordance with the algorithm [1]

$$t_k = \begin{cases} (k-1)\cdot a, \xi((k-1)\cdot a, X_{k-1}) \leq 0 \\ k \cdot a, \xi(t, X(t)) > 0, (k-1)\ a < t < k\ a \\ t_k \rightarrow \xi(t_k, X(t_k)) = 0, \\ \xi((k-1)\cdot a, X_{k-1}) > 0, \\ \xi(k \cdot a, X_k) < 0, k = 1,2,... \end{cases} \quad (4)$$

Where X_K and X_{K-1} are the values of the vector of state variables of the control system, taken in discrete time $k \cdot a$ and $(k-1)\cdot a$; the third line means that t_k is determined as the least root of the corresponding equation.

3. Descriptions of the results

As the subject of investigations is the D.C. traction motor drive with the pulse regulation of its current and PI-stage in the current regulator, whose equivalent scheme is shown in Fig.1. Using the physical principles of functioning of the system under consideration as the base we can present the system in the form of two interacted oscillators.

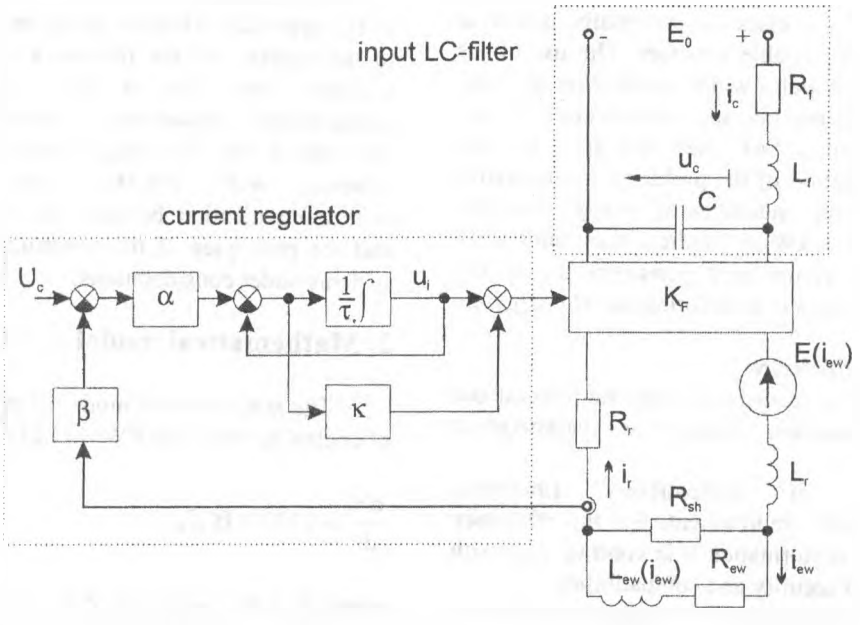


Fig. 1. The equivalent scheme of the D.C. traction motor drive with PWM

The first oscillator is the input LC-filter of the system and as the second oscillator may be considered the control system with PWM itself (the system with outer synchronizing impact, whose period is $T=a$) as a combination of the subject of regulation (of the motor), the regulating element (of the pulse converter) and the current regulator with PI-stage.

It is quite evident that the dynamics of such a system will be highly complicated. Here you may see both the phenomena cased by the interaction of two oscillators with irrational (in the general case) ratio of their frequencies and the phenomena which are due to the structure and the principle of operation of the regulator (especially the type and kind of modulation). To make a qualitative analysis of the regularities of the development of the dynamics of the system under consideration we shall carry on research in two stages. First, let us consider the control system with PWM-2 without taking into account the input LC-filter (that is, we proceed from the assumption that the input of the pulse element is connected to a source of unlimited power). The aim on consideration of such a simplified scheme is to reveal characteristic regularities of the development of the dynamics stipulated by the structure and the principles of operation of the regulator of the control system.

The mathematical model of the simplified system has the form

$$\frac{dX}{dt} = A \cdot X + B(K_F), \quad X = \{i_r, i_{ew}, u_i\}', \qquad (5)$$

$$A = \begin{vmatrix} -\frac{(R_r + R_{sh})}{L_r} & \frac{(R_{sh} - k_r n)}{L_r} & 0 \\ \frac{R_{sh}}{L_{ew}} & -\frac{(R_{sh} + R_{ew})}{L_{ew}} & 0 \\ \frac{\alpha\beta}{\tau} & 0 & -\frac{1}{\tau} \end{vmatrix},$$

$$B(1) = \begin{vmatrix} \frac{(E_0 - k_e n)}{L_r} \\ 0 \\ -\frac{\alpha U_c}{\tau} \end{vmatrix}, \quad B(0) = \begin{vmatrix} -\frac{k_e n}{L_r} \\ 0 \\ \frac{\alpha U_c}{\tau} \end{vmatrix},$$

$$\xi(t, X(t)) = \alpha \cdot (\kappa \cdot U_c - S' \cdot X(t)) - U_0 \cdot \left(\frac{t}{a} - E_1\left(\frac{t}{a}\right)\right),$$

$$S' = \left\{\kappa \cdot \beta, 0, \frac{1-\kappa}{\alpha}\right\},$$

where R_r, R_{sh}, R_{ew}, L_{ew}, L_r, α, β, τ, κ, U_c, U_0 - are the parameters of the elements of the control system, n is the velocity of the shaft of electric motor, E_0 is a power supply voltage. The nonlinear dependence of magnetic flux of the motor on the excitation winding current was approximated by the dependence $c_e\hat{O}(i_{ew}) = k_r \cdot i_{ew} + k_e$. The commutation of the pulse converter is accomplished in accordance with equations (3)-(4).

Let the parameters α and n to be varied. The rest parameters are assumed to be fixed and having the following values $R_r=0{,}0427\ \Omega$; $R_{sh}=0{,}615\ \Omega$; $R_{ew}=0{,}027\ \Omega$; $L_{ew}=0{,}98$ mH; $L_r=3{,}32$ mH; $\beta=0{,}01$; $\tau=0.01$ s; $\kappa=0.1$; $U_c=3$ V; $U_0=1{,}5$ V; $E_0=550$ V; $a=0{,}0025$ s, $k_r=0{,}0005$; $k_e=0{,}146$.

The analysis of the system (5) was made in the terms of the shift mapping [2].

$$X_k = C_k X_{k-1} + V_k, \qquad (6)$$

$$C_k = C^{(0)} C^{(1)}, \quad V_k = C^{(0)} V^{(1)} + V^{(0)},$$

$$C^{(1)} = e^{A \cdot a \cdot z_k}, \quad C^{(0)} = e^{A \cdot a \cdot (1 - z_k)},$$

$$V^{(1)} = -\left[E - C^{(1)}\right] A^{-1} B(1),$$

$$V^{(0)} = -\left[E - C^{(0)}\right] A^{-1} B(0), \quad k = 1, 2, \ldots .$$

The results of the analysis of the dynamics of the model (5) in the space of parameters $\Pi = \{\alpha, n;\ 3<\alpha<20,\ 200<n<2000\}$ are in Fig.2a as a two parametric bifurcation diagram. To understand these results better, in Fig.2b-2c we represent one parametric bifurcation diagrams built up for the sections $\alpha=7$ and $\alpha=18$ respectively. The domains of existence of different periodic motions are denoted in Fig.2a as Π with index, were the index determines the period ratio of the corresponding motion relative to the period of synchronizing impact of PWM (the number of m-cycle [1,2]). The domain of existence of chaotic motion is denoted as Π_{chaos} (see Fig.2a). The characteristic bifurcation curves are denoted as N with two indices (Fig.2a). Here the lower index "-" determines the curve of classic "mild" period doubling, and the lower index "c" determines the C-bifurcation curve [3]. The upper index of N determines the number of m-cycle that undergo a bifurcation. In Fig.2a Γ_0 denotes the curve of

break of the conditions of existence of periodic motions in the system.

Even superficial study of the diagram in Fig.2 enables to notice that, first, the domain of periodicity are alternated with the domains of chaotic motion and, second, the domains of periodicity are in the order of increasing of their periods in two times (see Fig.2a from below to upward). Such a superficial view would be sufficient for the continuous dynamic system to come to the conclusion, that the characteristic scenario of occurrence of chaotic motion is the scenario of period doubling. But for the discrete system (5) it is not sufficient. The main difference of the dynamics of the system (5) from "canonic" scenario of period doubling is the presence of C-bifurcation boundaries N_c in the plane Π. For more detailed study of bifurcations of dynamic motions of the system (5)

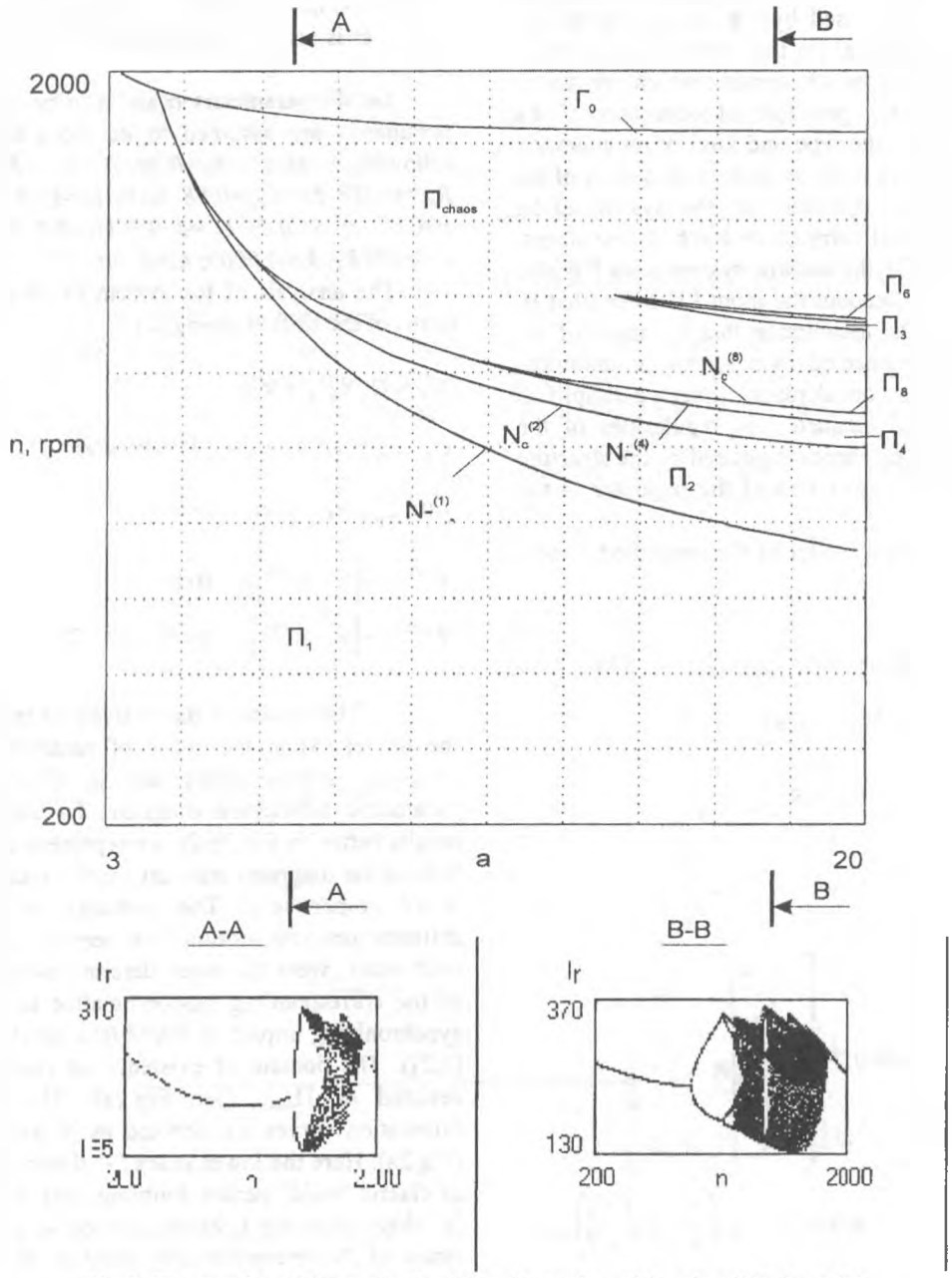


Fig.2. The results of the analysis of the dynamics of the simplified model.

Fig.3 shows a fragment of one parametric bifurcation diagram for the section $\alpha=9,1$; $1200\ rpm<n<1460\ rpm$ and the diagram of greatest multiplicator of the corresponding stationary motions. The values of the $i_r[k]$ $k=0,1,...$ (corresponding to the shift mapping) are plotted an the axis Y (see Fig.3a).The unstable stationary motions are marked by a dotted line in Fig.3.

Over the range $n < n_{-}^{(1)}, \left(\alpha = 9,1; n_{-}^{(1)}\right) \in N_{-}^{(1)}$ there exists a unique stable motion- 1-cycle. When n reaches the value $n_{-}^{(1)}$ the 1-cycle looses its stability as a result of bifurcation of period doubling and a stable 2-cycle appears in the

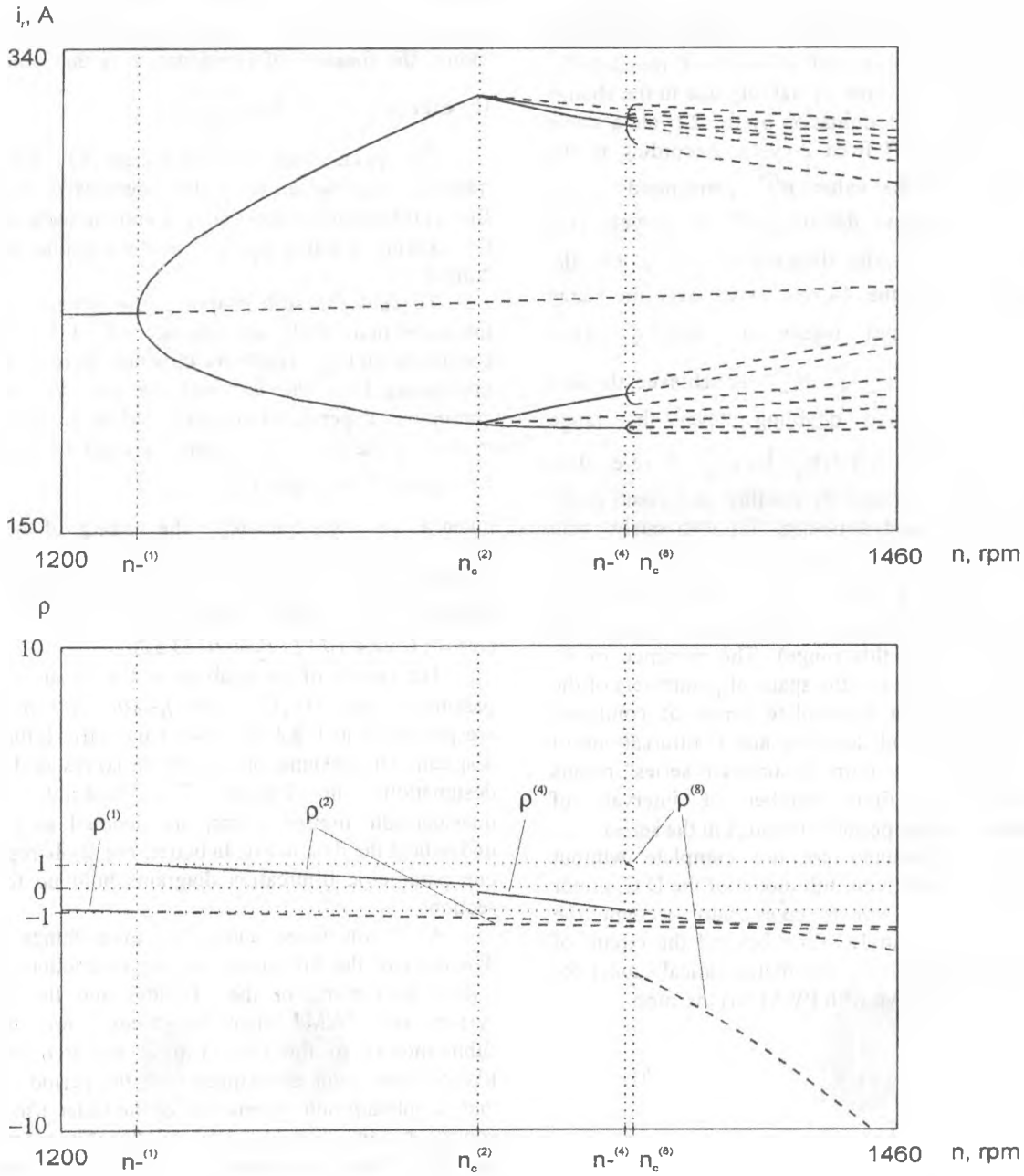


Fig.3. A fragment of one parametric bifurcation diagram and the diagram of greatest multiplicator of the corresponding stationary motions

system, which in its turn is a unique stable stationary motion in the system over a range $n_{-}^{(1)} < n < n_{c}^{(2)}, \left(\alpha = 9{,}1; n_{-}^{(2)}\right) \in N_{c}^{(2)}$. When $n = n_{c}^{(2)}$ is achieved, the domain of determination of the 2-cycle is changed (that is, the quantity of intervals of structure constancy of the system is changed which form a stationary motion) and the multiplicators of the 2-cycle are changed abruptly. In this case the greatest multiplicator becomes more than *-1* that determines the break of the conditions of stability of the 2-cycle. Simultaneously, with loss of stability due to the change of the domain of determination by the 2-cycle a stable 4-cycle branches off from 2-cycle. According to the classification [3] the value $n_{c}^{(2)}$ corresponds to C-bifurcation of period doubling of the 2-cycle (see boundary $N_{c}^{(2)}$ on the diagram of Fig.2). On the diagram of Fig.3a the 4-cycle exists over the range $n_{c}^{(2)}$ to $n_{-}^{(4)}$ and looses its stability when $n = n_{-}^{(4)}, \left(\alpha = 9{,}1; n_{-}^{(4)}\right) \in N_{-}^{(4)}$ is achieved through bifurcation of period doubling. Over the range $n_{-}^{(4)} < n < n_{c}^{(8)}, \left(\alpha = 9{,}1; n_{c}^{(8)}\right) \in N_{c}^{(8)}$ there exists a stable 8-cycle that looses its stability as a result of C-bifurcation of period doubling. Simultaneously, with loss of stability of the 8-cycle an unstable 16-cycle branches off from it. With $n > n_{c}^{(8)}$ there is no stable stationary motions in the system (a chaotic motion exist in the system over this range). The presence of C-bifurcation boundaries in the space of parameters of the system stipulates an incomplete series of combined bifurcations of period doubling and C-bifurcations of period doubling. The term "incomplete series" means that there is a finite number of intervals of corresponding stable periodic motions in the series.

The data obtained are not complete without consideration of relatively full model of the D.C. motor drive with PWM, which takes into account the influence of the input LC-filter beyond the circuit of regulation. The relatively full mathematical model for the electric motor drive with PWM has the form

$$\frac{dx_1}{dt} = -\frac{R_f}{L_f} x_1 - \frac{1}{L_f} x_2 + \frac{E_0}{L_f}, \qquad (6)$$

$$\frac{dx_2}{dt} = \frac{1}{C} x_1 - \frac{K_F(\xi)}{C} x_3,$$

$$\frac{dx_3}{dt} = -\frac{R_r + R_{sh}}{L_r} x_3 + \frac{R_{sh}}{L_r} x_4 + \frac{K_F(\xi)}{L_r} x_2 - \frac{E(x_4)}{L_r},$$

$$\frac{dx_4}{dt} = \frac{R_{sh}}{L_{ew}(x_4)} x_3 - \frac{R_{ew} + R_{sh}}{L_{ew}(x_4)} x_4,$$

$$\frac{dx_5}{dt} = \frac{\alpha \cdot \beta}{\tau} x_3 - \frac{1}{\tau} x_5 - \frac{\alpha \cdot U_c}{\tau},$$

$$E(x_4) = n c_e \Phi(x_4), \; L_{ew}(x_4) = 2 \cdot p \cdot w \frac{\partial \Phi(x_4)}{\partial x_4},$$

where $X = \{i_c, u_c, i_r, i_{ew}, u_l\}^{'}$ the vector of state variables; p, w are the constructive parameters of the motor; the function of commutation is the same as in (5) with $S' = \left\{0{,}0, \kappa \cdot \beta, 0, \frac{1-\kappa}{\alpha}\right\}$

The specific feature of the system (6) is the use of nonlinear representation of dependencies of magnetic flux of the electric motor $c_e\Phi(i_{ew})$ and the inductance of the existing winding $L_{ew}(i_{ew})$ on the existing winding current.

To find the shift mapping the system (6) was integrated numerically over the intervals of the structure constancy and the solutions obtained were combined proceeding from the fact that the state vector is in continuous dependence on time. While analyzing the model (6) the parameter *n* and the value of resonance frequency of the input filter $f_r = 1/\left(2 \cdot \pi \cdot \sqrt{L_f \cdot C}\right)$ are taken as varying parameters. The value f_r of the input filter was varied by change of L_f over the range $[0{,}066 \cdot 10^{-3}$ to $4{,}2 \cdot 10^{-3}]$ H. The rest parameters were assumed to be fixed as well as for the simplified model ($\alpha=10$; $C=2{,}4 \cdot 10^{-3}$ F; $R_f=0{,}0123$ Ω).

The results of the analysis of the dynamic in the parameter space $\Pi=\{f_r, n;\ 50< f_r<400,\ 200<n<2000\}$ are presented in Fig.4 as a two parametric bifurcation diagram. The designations in Fig.4a correspond to the designations in Fig.2a. The domains, where quasiperiodic motion exists, are denoted as Π_q. To understand the data in Fig.4a better, Fig.4b-4c represent one parametric bifurcation diagrams built up for two sections.

As it was noted above, the main things in the dynamics of the full model are the interaction of two related oscillators: of the LC-filter and the control system with PWM. When the values f_r are close to subharmonics of the ratio 1/m of the frequency of PWM, there occur oscillations with the period $T=m\,a$, that is, subharmonic resonances of the order 1/m in the control system. The irrational relationship between f_r and the timing frequency of PWM leads to the appearance of extensive domains of existence of quasiperiodic motions (toroidal manifolds) Π_q in the plane Π. In spite of the fact that these domains are

shown in Fig.4 as uniform while studying them in detail one can observe numerous narrow "windows" of synchronization which are characterized by the ratio of f_r to the frequency of PWM as rational fraction p/q (resonance p/q). The periodic motions from the "windows" of synchronization can undergo numerous bifurcations (and C-bifurcations, in particular) which lead to the appearance of chaotic motion, as well as one can observe the loss of synchronization, etc. As the value of the domains of resonance p/a is quite small and the regularities of change of dynamic motions in resonance domains typical, the paper analyze only the domains of subharmonic resonances of the order 1/m.

In Fig.4 the domains of subharmonic resonances of the order 1/m (Π_2, Π_3, Π_4, Π_5, etc.) are shown in gray color. These domains Fig.4a are located in some vicinity of corresponding values f_r (200 Hz, 133 Hz, 100 Hz, 80 Hz etc.) and can shift slightly with variation in the parameters of the control system. The width of the domain of subharmonic resonance number m Δf_m is not equal to zero for any number m of resonance (it is stipulated by dissipativeness of the system) and becomes less with increase in m.

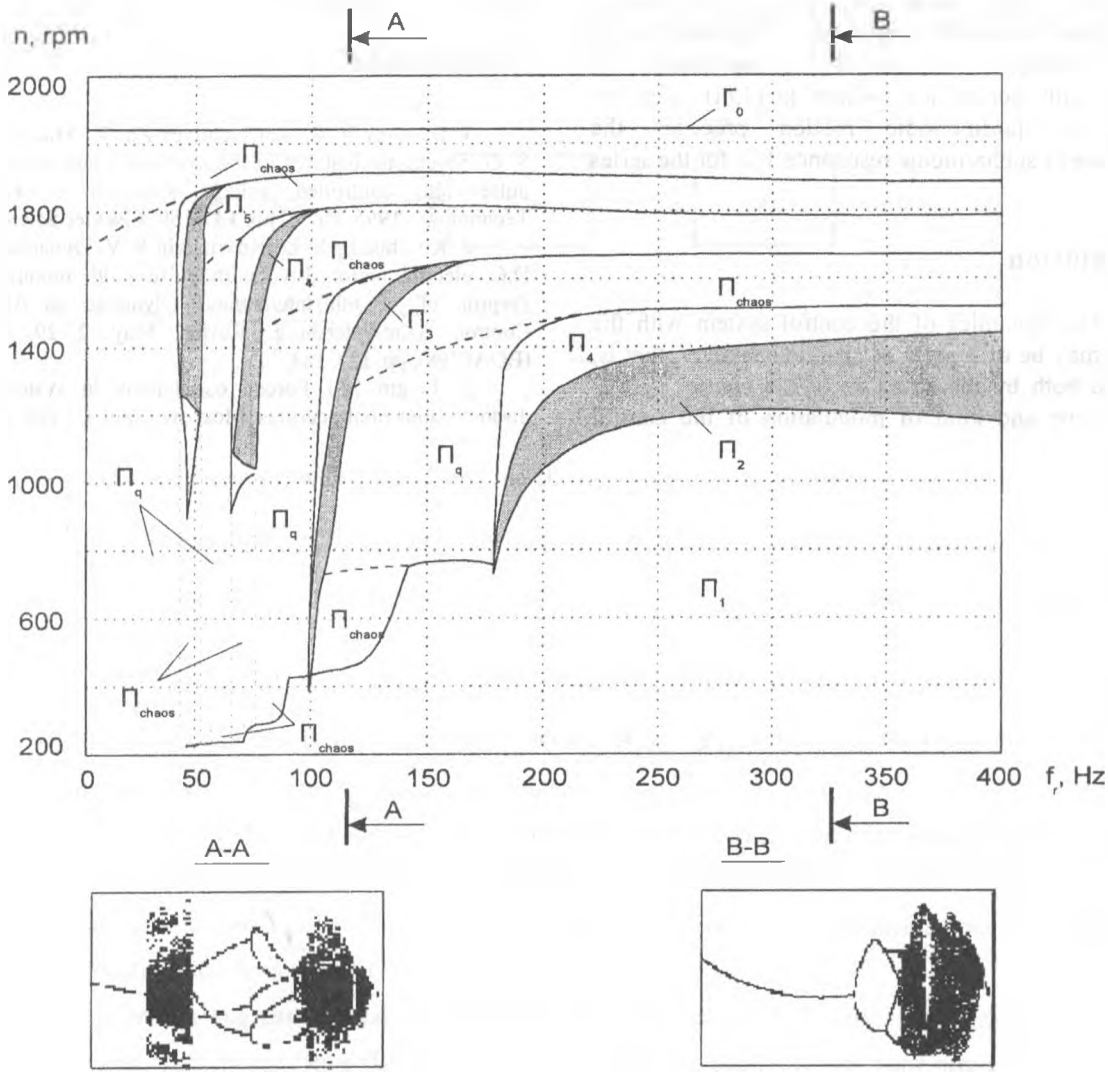


Fig.4. The results of the analysis of the dynamics of the full model.

The quasistatic growth of n from the initial point in the domain of the preset periodic motion Π_1 for a fixed f_r in the vicinity of the domain of subharmonic resonance 1/m leads to the combined series of bifurcations of period doubling and C-bifurcations of period doubling, which can be represented a symbolical series

$$a \to m\,a \to 2\,m\,a \to 4\,m\,a \to \ldots, \quad (7a)$$
$$a \to \ldots \to m\,a \to 2\,m\,a \to 4\,m\,a \to \ldots. \quad (7b)$$

The analysis of simplified model shown that these series may be incomplete, that is, determined by the revealed regularities of occurrence of C-bifurcation boundaries in the parameter space of the control system. The characteristic property of the series of the stationary motions (7a) (or 7b) is the absence of motions with period k a, where $k \in (1,m)$. Besides, chaotic or quasiperiodic motion precedes the appearance of subharmonic resonance 1/m for the series (7b).

4. Conclusion

1. The dynamics of the control system with the PWM-2 may be of regular or chaotic character, that is, stipulated both by the structure of the control system and the type and kind of modulation of the control system. In particular, the peculiarity of the structure of the considered D.C. electric drive with PWM-2 is the presence of two interacted oscillations - the input LC-filter and the control system with PWM (the oscillator with outer synchronizing impact). In this case one of the scenarios of chaotization of motions in the control system under consideration is the disturbance of synchronized resonance motions on toroidal manifolds.

2. The characteristic scenario of chaotization of oscillations in the control system with PWM-2 is a combined series of a bifurcations of period doubling and C-bifurcations of period doubling. It is find out, that this series may be incomplete, that is, stipulated by C-bifurcation phenomena.

5. References

1. Baushev V. S., Zhusubaliyev Zh. T., Mikhal`chenko S. G. Stochastic features in the dynamic characteristics of a pulse-width controlled voltage stabilizer. // Electrical Technology, 1996, No. 1, pp. 137-150, Elsevier Science.

2. Koschinsky S. L., Kovrizhkin S. V. Dynamics of the D.C. electric motor drive with pulse-width modulation. // Preprint of the 6th International Olympiad on Automatic Control, Saint-Petersburg, Russia, May 27-29, 1998 - (BOAC'98), pp. 138-144.

3. Feigin M.I Forced oscillations in systems with discontinuous nonlinearities. Moscow, Nauka, 1994, pp. 288.

Dynamic Algorithms of Multilayered Neural Networks Training in Generalized Training Plant

Efimov D.V., Zakharenkova T.A., Terekhov V.A., Tyukin I.Yu.
The Department of Automatics and Control Processes
St. Petersburg State Electrical Engineering University "LETI"
Phone: 234-64-35, e-mail: efde@mail.rcom.ru

Abstract - *The new modifications of multilayered neurak networks training algorithms in a generalized training plant structure are introduced. The first modification for algorithm " on error backpropagation algorithm through time " is introduced and its sufficient conditions of a training procedure stability are obtained. Other modification for speed gradient algorithm of an error backpropagation is obtained , where measurement state vector in a training procedure instead of a parametrical model of plant is used.*

1. INTRODUCTION

It is known that a connection of a neurocontroller with a control plant (CP) may be represented as generalized training plant (GTP) (fig. 1) [1,2].

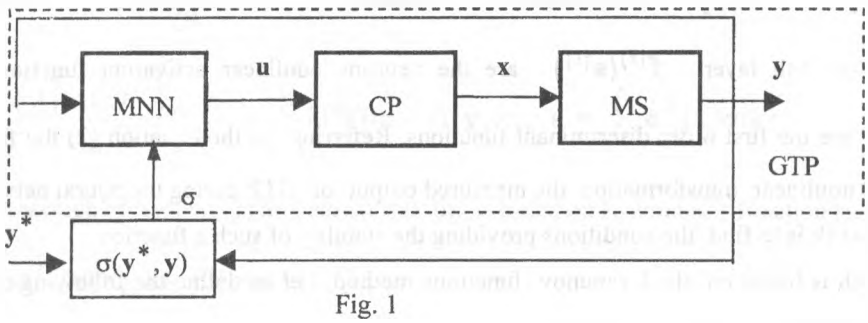


Fig. 1

There MS is a measuring system, which translates the statement vector $\mathbf{x}$ to the measured vector $\mathbf{y} = \mathbf{h}(\mathbf{x}(t))$, where $\mathbf{h}(\cdot)$ is a vector function; $\mathbf{y}^*$ is a desired function, $\boldsymbol{\sigma} = \boldsymbol{\sigma}(\mathbf{y}^*, \mathbf{y})$ is a generalized error vector, which is used for network training. The architecture of MNN is defined by $N^K_{n_0, n_1, \ldots n_K}$ symbol, where K is an amount of network layers, n_0 is a number of inputs, n_i, $i \in \overline{1, K}$ is a count of the artificial neurons or basic processor elements (BPE) in the i-th layer. The problem how to find the control function $\mathbf{u}(\mathbf{y}^*, \mathbf{y})$, which provides a minimum of function Q, is reduced in this case to the neural network training problem.

The base algorithm of multilayered neural network training is the algorithm of an error backpropagation (BP). Its direct use for synthesis of neural network training algorithms in problems of dynamic plants control by virtue of a series of the reasons is hampered [1]. However it forms the basis for synthesis of multilayered neural network training algorithms modifications in a structure of GTP.

In this paper we suggest the training algorithm for MNN within GTP (fig. 1), which is based on error backpropagation algorithm through time (BPTT) [3]. Also the algorithm resulted from back propagation error method and speed-gradient method (SG) [4] and was later called speed back propagation error algorithm (SBP) are introduced. Using that algorithm one can take into account the dynamics of the network training process and to cover a wide spectrum of problems. Unfortunately the algorithm lacks an adequate control plant model, it also takes a long time of computation and the necessity first to make the prognose of signal in the plant output. Modification of the algorithm

SBP, free from enumerated shortages is offered below.

2. BPTT ALGORITHM

We make an algorithm step estimation, which provides training process stability for functional $Q = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(k)\boldsymbol{\sigma}(k)$ at discrete-time point $k\Delta t$, $\Delta t = \text{const}$; $k = 1,2, \ldots$, and also a training error is defined by us as $\boldsymbol{\sigma}(k) = \mathbf{y}^*(k) - \mathbf{y}(k)$.

Let a plant be defined by the following equation:

$$\begin{cases} \mathbf{x}(k+1) = \mathbf{f}(\mathbf{x}(k), \mathbf{u}(k)); \\ \mathbf{y}(k) = \mathbf{h}(\mathbf{x}(k)). \end{cases} \quad (1)$$

The control vector on fig. 1 is found by MNN during its training:

$$\mathbf{u} = \mathbf{q}^{(K)} = \mathbf{f}^{(K)}(\mathbf{W}^{(K)}\mathbf{f}^{(K-1)}(\mathbf{W}^{(K-1)}\mathbf{f}^{(K-2)}(\ldots\mathbf{W}^{(\ell)}\mathbf{f}^{(\ell-1)}(\mathbf{W}^{(\ell-1)}\mathbf{f}^{(\ell-2)} \times (\cdots\mathbf{W}^{(2)}\mathbf{f}^{(1)}(\mathbf{W}^{(1)}\mathbf{y}(k-1))\cdots)\cdots))) = \mathbf{F}(\mathbf{y}), \quad (2)$$

where $\mathbf{W}^{(\ell)} = \left[\mathbf{w}_1^{(\ell)} : \ldots : \mathbf{w}_{n_\ell}^{(\ell)}\right]^T$ is the weight-factor matrix for layer $\ell \in \overline{1,K}$, $\mathbf{w}_i^{(\ell)}$ is a weight-factor vector for the i-th BPE in the l-th layer, $\mathbf{f}^{(\ell)}(\mathbf{s}^{(\ell)})$ are the neurons nonlinear activation functions in the l-th layer, $\mathbf{s}^{(\ell)} = \mathbf{w}^{(\ell)}\mathbf{q}^{(\ell-1)}$ are the first order discriminant functions. Referring to the equation (2) the function carries out a complex dynamical nonlinear transformation the measured output of GTP during the neural network training process. The subject of this work is to find the conditions providing the stability of such a function.

The research is based on the Lyapunov functions method. Let us define the following training algorithm for MNN in a GTO structure:

$$\delta\mathbf{w}_i^{(\ell)}(n) = -\gamma \sum_{j=1}^{N} \frac{\partial \boldsymbol{\sigma}^T(j)\boldsymbol{\sigma}(j)}{\partial \mathbf{u}(1)} \frac{\partial \mathbf{u}(1)}{\partial \mathbf{w}_i^{(\ell)}}, \quad (3)$$

where γ is an algorithm step, n is a number of steps. The partial derivative from equation (3) may be calculated by following formula:

$$\frac{\partial \boldsymbol{\sigma}^T(j)\boldsymbol{\sigma}(j)}{\partial \mathbf{u}(1)} = \frac{\partial \boldsymbol{\sigma}^T(j)\boldsymbol{\sigma}(j)}{\partial \mathbf{y}(j)} \frac{\partial \mathbf{y}(j)}{\partial \mathbf{x}(j)} \left(\frac{\partial \mathbf{f}(\mathbf{x}(j-1), \mathbf{u}(j-1))}{\partial \mathbf{u}(j-1)} \frac{\partial \mathbf{u}(j-1)}{\partial \mathbf{y}(j-1)} + \frac{\partial \mathbf{f}(\mathbf{x}(j-1), \mathbf{u}(j-1))}{\partial \mathbf{x}(j-1)} \frac{\partial \mathbf{x}(j-1)}{\mathbf{y}(j-1)} \right) \cdots \frac{\partial \mathbf{x}(2)}{\partial \mathbf{u}(1)}. \quad (4)$$

The parity such as (4) is used in error backpropagation algorithm (BP). However, this "error backpropagation algorithm" differs from BP algorithm, because the described algorithm is not passed from layer to layer, it's passed from state to state, i. e. from one time moment to next. That is why this algorithm is called "error backpropagation through time".

As any recurrent algorithm the gradient procedure (3) may be stable or unstable depending on the algorithm parameters and, first of all, on the algorithm step γ, when the criterion is the function $Q = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(k)\boldsymbol{\sigma}(k)$. Besides this in practice one requires to provide the MNN training process stability only if the desired function, network architecture and the initial state are changed. In the square training criterion $Q = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(k)\boldsymbol{\sigma}(k)$ case a sufficient conditions follow from the theorem.

Theorem. *Let the functions* $\mathbf{f}, \mathbf{h} \in \mathbf{C}^1$ *in system (1) and there exist: a) partial derivatives* $\dfrac{\partial \mathbf{x}(j)}{\partial \mathbf{y}(j-1)}$

b) vector function $\delta\mathbf{y}(\mathbf{W}, k)$ *which may be represented in form (5) and also indicate an increment of the vector* $\mathbf{y}(\mathbf{W}, k)$ *followed a variation of weight-factor matrix* $\mathbf{W}$.

$$\delta\mathbf{y}(\mathbf{W}, k) = \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \frac{\partial \mathbf{y}(\mathbf{W}, k)}{\partial w_{i,j}^{(\ell)}} \delta w_{i,j}^{(\ell)} + \mathbf{e}(\delta\ \mathbf{W}, k) \tag{5}$$

Then the gradient procedure (3) convergence for square criterion Q *if the following conditions are satisfied:*

1) there exists $\lambda(\mathbf{W}, \mathbf{y}, k) > 0$ *such as*

$$\delta\mathbf{y}(\mathbf{W}, k)^T \delta\mathbf{y}(\mathbf{W}, k) \le \lambda(\mathbf{W}, \mathbf{y}, i) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \left(\delta w_{i,j}^{(\ell)}\right)^2;$$

2) $\gamma \in \left(\dfrac{1 - \sqrt{1 - 8\tilde{\lambda}\varepsilon\alpha^{-1}}}{2\tilde{\lambda}}, \dfrac{1 + \sqrt{1 - 8\tilde{\lambda}\varepsilon\alpha^{-1}}}{2\tilde{\lambda}} \right)$ *in the case* $1 - 8\tilde{\lambda}\varepsilon\alpha^{-1} \ge 0$ *and* $\gamma = 0$ *in the case* $1 - 8\tilde{\lambda}\varepsilon\alpha^{-1} < 0$,

where $\alpha = \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \left(\dfrac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right)^2$, $\tilde{\lambda} = \sum_{k=1}^{N} \lambda(\mathbf{W}(n-1), \mathbf{y}, k)$, $\varepsilon = \sum_{k=1}^{N} \mathbf{e}^T(k)\boldsymbol{\sigma}(k)$.

Proof. The research of system (3) movement stability is based on Lyapunov's function analysis:

$$V(n) = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(\mathbf{W}(n), k)\boldsymbol{\sigma}(\mathbf{W}(n), k) - Q_{\min} \tag{6}$$

where $Q_{\min}$ is minimum value of a functional Q at neural network architecture $N_{n_0, n_1, \ldots n_K}^{K}$ and activation functions $f(\cdot)$.

Let us consider an increment $\Delta V(n)$:

$$\Delta V(n) = V(n) - V(n-1) = \sum_{k=1}^{N} \boldsymbol{\sigma}^T(\mathbf{W}(n), k)\boldsymbol{\sigma}(\mathbf{W}(n), k) - \boldsymbol{\sigma}^T(\mathbf{W}(n-1), k), \boldsymbol{\sigma}(\mathbf{W}(n-1), k). \tag{7}$$

The generalized error $\boldsymbol{\sigma}(\mathbf{W}(n), i)$ in (7) is represented in the following kind:

$$\boldsymbol{\sigma}(\mathbf{W}(n), k) = \mathbf{y}^{(*)}(k) - \mathbf{y}(n, k) = \mathbf{y}^{(*)}(k) - (\mathbf{y}(\mathbf{W}(n-1), k) + \delta\mathbf{y}(\mathbf{W}(n-1), k)), \tag{8}$$

where $\delta\mathbf{y}(\mathbf{W}(n-1), k)$ is a vector function of the $\mathbf{y}(\mathbf{W}(n-1), k)$ vector increment which is determined by the weight-factors variation. Then the Lyapunov's function (6) increment ΔV referred to (7), (8) is discovered as:

$$\begin{aligned} \Delta V(n) = \sum_{k=1}^{N} \Big((\mathbf{y}^{(*)}(k) - (\mathbf{y}(\mathbf{W}(n-1), k) + \delta\mathbf{y}(\mathbf{W}(n-1), k))^T (\mathbf{y}^{(*)}(k) - -(\mathbf{y}(\mathbf{W}(n-1), k) + \delta\mathbf{y}(\mathbf{W}(n-1), k)) \Big) - \\ - \sum_{k=1}^{N} (\mathbf{y}^{(*)}(k) - \mathbf{y}(\mathbf{W}(n-1), k))^T (\mathbf{y}^{(*)}(i) - \mathbf{y}(\mathbf{W}(n-1), k)). \end{aligned} \tag{9}$$

Simple transformations of equation (9) result its in the following form:

$$\Delta V(n) = \sum_{k=1}^{N} \delta\mathbf{y}(\mathbf{W}(n-1), k)(2(\mathbf{y}(\mathbf{W}(n-1), k)) - \mathbf{y}^{(*)}(\mathbf{W}(n-1), k)) + \delta\mathbf{y}(\mathbf{W}(n-1), k)). \tag{10}$$

It is necessary to emphasize that the increment $\delta\mathbf{y}(\mathbf{W}(n-1), k)$ in equation (10) may be represent in the kind of following sum:

$$\delta\mathbf{y}(\mathbf{W}(n-1),k) = \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}} \frac{\partial \mathbf{y}(\mathbf{W}(n-1),k)}{\partial w_{i,j}^{(\ell)}} \delta w_{i,j}^{(\ell)}(n-1) + \mathbf{e}(\delta\mathbf{W},\mathbf{y},k), \tag{11}$$

where $\delta w_{i,j}^{(\ell)} = (-\gamma(n-1))\dfrac{\partial Q}{\partial w_{i,j}^{(\ell)}}$. Then let us write down the increment ΔV of the chosen Lyapunov's function (6) refer to (10), (11):

$$\Delta V(n) = \left(\sum_{k=1}^{N} \delta\mathbf{y}(\mathbf{W}(n-1),k)^T \delta\mathbf{y}(\mathbf{W}(n-1),k) + 2\mathbf{e}(\delta\mathbf{W},\mathbf{y},k)^T \boldsymbol{\sigma}(\mathbf{W},\mathbf{y},k)\right) + \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}} \left(\delta w_{i,j}^{(\ell)}\right)^2 (-\gamma(n-1))^{-1} \le$$
$$\le \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}} \left(\delta w_{i,j}^{(\ell)}\right)^2 \left((-\gamma(n-1))^{-1} + \sum_{k=1}^{N}\lambda(\mathbf{W},\mathbf{y},k)\right) + 2\sum_{k=1}^{N}\mathbf{e}(\delta\mathbf{W},\mathbf{y},k)^T\boldsymbol{\sigma}. \tag{12}$$

Let $\alpha = \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\left(\dfrac{\partial Q}{\partial w_{i,j}^{(\ell)}}\right)^2 = \sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\left(\delta w_{i,j}^{(\ell)}\right)^2(\gamma(n-1))^{-2}$, $\tilde{\lambda} = \sum_{k=1}^{N}\lambda(\mathbf{W}(n-1),\mathbf{y},k)$, $\varepsilon = \sum_{k=1}^{N}\mathbf{e}^T(k)\boldsymbol{\sigma}(k)$.

Then the equation (12) can be rewritten as $(-\gamma)^2\tilde{\lambda}\alpha - \gamma\alpha + 2\varepsilon \le 0$. If γ is expressed we receive that:

$\gamma \in \left(\dfrac{1-\sqrt{1-8\tilde{\lambda}\varepsilon\alpha^{-1}}}{2\tilde{\lambda}}, \dfrac{1+\sqrt{1-8\tilde{\lambda}\varepsilon\alpha^{-1}}}{2\tilde{\lambda}}\right)$ in the case $1-8\tilde{\lambda}\varepsilon\alpha^{-1} \ge 0$ and $\gamma = 0$ in the case $1-8\tilde{\lambda}\varepsilon\alpha^{-1} < 0$.

The theorem is proved.

It can be shown that the function $\lambda(\mathbf{W},\mathbf{y},k)$ which meets the theorem condition *1)* under condition *2)* always exists. To obtain this it will be enough to consider the variation rate of a left bound γ parameters (to refer to the condition *2)*) for small increment $\delta w_{i,j}^{(\ell)}$. It is necessary to emphasize, that in a general case the stability by Lyapunov's method and, hence, the negative determinacy of Lyapunov function do not guarantee the stability of the gradient procedure (3). However, when we choose a suitable Lyapunov's function V, which equals the MNN training functional Q accurate to any constant (as it is shown in the proof), the function V negative determinacy provides non-growing functional Q variations.

It is following from the theorem condition *1)* that the function may be defined by the following inequality:

$$\delta\mathbf{y}(\mathbf{W},k)^T\delta\mathbf{y}(\mathbf{W},k)\left(\sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\left(\delta w_{i,j}^{(\ell)}\right)^2\right)^{-1} \le \lambda(\mathbf{W},\mathbf{y},k) \tag{13}$$

Let symbol $\mathbf{J}(h)$ is:

$$\mathbf{J}(h) = \left(\prod_{\nu=3}^{h}\frac{\partial\mathbf{y}(\nu)}{\partial\mathbf{y}(\nu-1)}\right)\frac{\partial\mathbf{y}(2)}{\partial\mathbf{x}}\frac{\partial\mathbf{x}(2)}{\partial\mathbf{q}^{(K)}(1)} \tag{14}$$

When we discover the $\delta\mathbf{y}(\mathbf{W},k)$ increment in (13) refer to (14) we'll receive:

$$\left(\sum_{h=2}^{k}\mathbf{J}(h)\sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\frac{\partial\mathbf{q}^{(K)}(1)}{\partial w_{i,j}^{(\ell)}}\delta w_{i,j}^{(\ell)} + \mathbf{e}(\delta\mathbf{W},\mathbf{y},h)\right)^T \times \left(\sum_{h=2}^{k}\mathbf{J}(h)\sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\frac{\partial\mathbf{q}^{(K)}(1)}{\partial w_{i,j}^{(\ell)}}\delta w_{i,j}^{(\ell)} + \mathbf{e}(\delta\mathbf{W},\mathbf{y},h)\right)\left(\sum_{\ell=1}^{K}\sum_{i=1}^{n_\ell}\sum_{j=1}^{n_{\ell-1}}\left(\delta w_{i,j}^{(\ell)}\right)^2\right)^{-1} \le \lambda(\mathbf{W},\mathbf{y},k). \tag{15}$$

The inequality (15) allows to estimate bounds of the γ parameters "stable" area. In particular, if we neglect the function $\mathbf{e}(\delta\mathbf{W},\mathbf{y},h)$ values because of the increments $\delta w_{i,j}^{(\ell)}$ are rather small (as far as the function $\mathbf{e}(\delta\mathbf{W},\mathbf{y},h)$ is $o(\delta\mathbf{W})$) we'll obtain the following step γ estimation:

$$\gamma \le \sum_{k=2}^{N} \left(\left(\sum_{h=2}^{k} \mathbf{J}(h) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \frac{\partial \mathbf{q}^{(K)}(1)}{\partial w_{i,j}^{(\ell)}} \frac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right)^T \times \left(\sum_{h=2}^{k} \mathbf{J}(h) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \frac{\partial \mathbf{q}^{(K)}(1)}{\partial w_{i,j}^{(\ell)}} \frac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right) \right)^{-1} \left(\sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \left(\frac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right)^2 \right). \quad (16)$$

We should like to emphasize that the theorem conditions are sufficient for the gradient procedure (3). Also their observation provides roughness of the network training algorithm. Besides this function $\lambda(\mathbf{W}, \mathbf{y}, k)$ can be used for estimation of the MNN architecture and single neuron activation function influence the BPTT procedure stability.

The obtained expressions may be used for *adaptive* algorithm of MNN training within GTP because the function $\gamma(n)$ is corrected (varied) during the training process referring plant state, the amounts of MNN layers, neurons in the layer, the network initial conditions, the current weight-factor value and to the activation function in covert form. In other words this function depends on the plant's dynamics, the MNN architecture and state.

Let us consider an example.

Let a plant is defined by the following equation system:

$$\begin{cases} x_1(k+1) = x_1(k) + (-x_2(k) - x_3(k) + u_1(k))\Delta t \\ x_2(k+1) = x_2(k) + (x_1(k) + 0.3x_2(k) + u_2(k))\Delta t \\ x_3(k+1) = x_3(k) + (0.38x_1(k) - 4.5x_2(k) + x_1(k)x_3(k) + u_3(k))\Delta t, \end{cases}$$

where Δt is a discretization step. Let all components of the state vector are measured. Hence, the Jacobian $\frac{\partial \mathbf{x}(j)}{\partial \mathbf{y}(j-1)}$ used in BPTT algorithm exists:

$$\frac{\partial \mathbf{x}(j)}{\partial \mathbf{x}(j-1)} = \begin{pmatrix} 1 & -\Delta t & -\Delta t \\ \Delta t & 0.3\Delta t + 1 & 0 \\ (0.38 + x_3)\Delta t & -4.5\Delta t & x_1 \Delta t + 1 \end{pmatrix} + \begin{pmatrix} \Delta t & 0 & 0 \\ 0 & \Delta t & 0 \\ 0 & 0 & \Delta t \end{pmatrix} \frac{\partial \mathbf{u}(j-1)}{\partial \mathbf{x}(j-1)}$$

A reference function is set as $\mathbf{y}^*(t) = col(e^{-t}, e^{-t}, e^{-t})$ function.

To simulat the process we chose some MNN with a $N^3_{3,50,50,3}$ architecture and the sigmoidal activation functions $f(s) = \frac{1 - e^{-2s}}{1 + e^{-2s}}$ The conditions of training simulation was: N = 30 trajectory points, Δt =0.01c descretization step, $x_1(1) = x_2(1) = x_3(1) = 1$ initial conditions. The estimation of γ parameter is chosen as in the inequality (6). Let us mark:

$$\mathbf{D}(\ell) = \prod_{\ell_1 = K}^{\ell+1} \mathbf{w}^{(\ell_1)}; \quad (17)$$

$$d_{j,i}^{(\ell)} = \delta(j,i) \max(f_i'^{(\ell)}) \left| q_j^{(\ell-1)} \right|, \quad (18)$$

where $\delta(j,i)$ is Kronecer's symbol, $j = \overline{1, n_{\ell-1}}$, $i = \overline{1, n_\ell}$. After the partial derivations $\frac{\partial Q}{\partial w_{i,j}^{(\ell)}}$ were discovered in (16) we receive refer to (17), (18):

$$\gamma \le \sum_{k=2}^{N} \left(\left(\sum_{h=2}^{k} \mathbf{J}(h) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \left| \mathbf{D}(\ell) \mathbf{d}_i^{(\ell)} \right| \left(\mathbf{D}(\ell) \mathbf{d}_i^{(\ell)} \right)^T \boldsymbol{\sigma} \right)^T \times \left(\sum_{h=2}^{k} \mathbf{J}(h) \sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \left| \mathbf{D}(\ell) \mathbf{d}_i^{(\ell)} \right| \times \left(\mathbf{D}(\ell) \mathbf{d}_i^{(\ell)} \right)^T \boldsymbol{\sigma} \right) \right)^{-1} \times \left(\sum_{\ell=1}^{K} \sum_{i=1}^{n_\ell} \sum_{j=1}^{n_{\ell-1}} \left(\frac{\partial Q}{\partial w_{i,j}^{(\ell)}} \right)^2 \right). \quad (19)$$

These parameter γ estimations satisfy the theorem conditions when the variations $\delta w_{i,j}^{(\ell)}$ are small. Besides it can be shown that the weight-factors $\left|w_{i,j}^{(\ell)}\right| > w_{i,j\,\text{bound}}^{(\ell)}$ exist for MNN with sigmoidal activation functions and they provide a decrement of the criterion $Q(n)$ value.

The computer simulation showed that neural network training procedure has an undiverging reference. In fig. 2 a, b accordingly you can see the graphs which illustrates the variation of the algorithm step $\gamma(n)$ and training functional $Q(n)$ in the cases when estimation (19) is used and when $\gamma = \text{const} = 0.0015$.

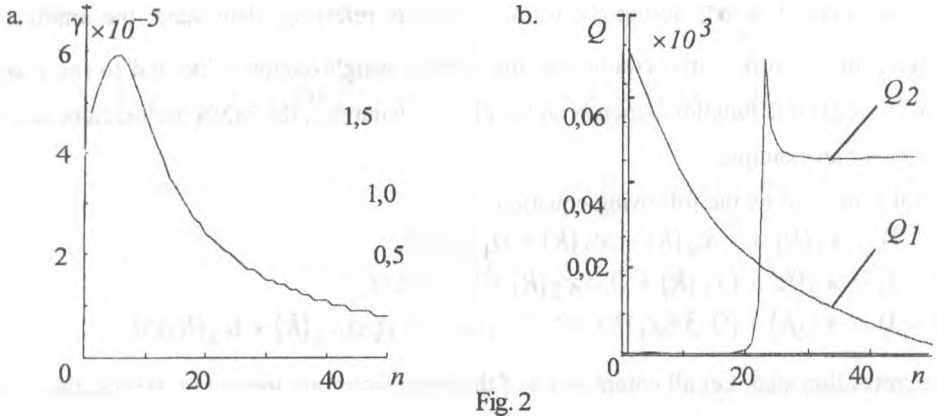


Fig. 2

This graphs allow to conclude that the computer simulation does not to contradict the theoretical items of the work.

3. SBP ALGORITHM

Let the dynamics of control plant is described by equations:

$$\mathbf{x} = \mathbf{f}(\mathbf{x}, \boldsymbol{\theta}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \quad \mathbf{x} \in R^n, \quad \mathbf{u} \in R^m. \tag{20}$$

The behaviour of control plant (20) depends on unknown parameters values $\boldsymbol{\theta} \in \Omega_{\boldsymbol{\theta}}$. The goal of control presented as desirable dynamic characteristics of control plant based on a reference model described by a system of linear differential equations:

$$\mathbf{x}^{\mathcal{M}} = \mathbf{A}^{\mathcal{M}}\mathbf{x}^{\mathcal{M}} + \mathbf{B}^{\mathcal{M}}\mathbf{r}, \tag{21}$$

where $\mathbf{x}^{\mathcal{M}} \in R^n$ is a state vector of a reference model, $\mathbf{A}^M, \mathbf{B}^M$ are model matrixes chosen so that a response of state vector $\mathbf{x}^{\mathcal{M}}$ on assigning action $\mathbf{r}$ carried a desirable character.

Let a control vector $\mathbf{u} = \mathbf{q}^{(K)}$ be the output of the last K th layer of a network. It is required to derive network training algorithm, which would change the vector of adjusted parameters $\mathbf{w}_i^{(\ell)}$ (where $\ell = \overline{1, K}$ - number of MNN layer, and $i = \overline{1, n_\ell}$ - number in a layer) to ensure reaching control goal at any vector of unknown parameters $\boldsymbol{\theta} \in \Omega_{\boldsymbol{\theta}}$.

A speed-gradient algorithm in the combined form [5] for set-up vectors $\mathbf{w}_i^{(\ell)}$ can be written as follows:

$$\frac{d\left(\mathbf{w}_i^{(\ell)} + \psi\right)}{dt} = -\Gamma \nabla_{\mathbf{w}} \omega(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t), \quad i = \overline{1, n_\ell}, \tag{22}$$

where $\omega(\,)$ is function, continuously differentiable on components of a vector $\mathbf{w}_i^{(\ell)}$; $\Gamma = \Gamma^T > 0$ is $((n_\ell + 1) \times (n_\ell + 1))$ gain matrix; ψ is a pceudo-gradient function which meets to condition $\| \psi\, \omega(\cdot) \| \geq 0$. The stability theorems for speed-gradient systems (20),(22) can be found in [5].

For quasistationary plant the function $\omega(\cdot)$ is determined as follows:

$$\omega(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t) = \left[\mathbf{f}(\mathbf{x}, \boldsymbol{\theta}) + \mathbf{g}(\mathbf{x})\mathbf{u}(\mathbf{w}_i^{(\ell)})\right]^T \nabla_{\mathbf{x}} J(\mathbf{x}; t) + \mathbf{x}^{\mathcal{M}^T} \nabla_{\mathbf{x}^{\mathcal{M}}} J(\mathbf{x}; t). \tag{23}$$

Instead of (22) it is possible to use SG algorithm in its finite form:

$$\mathbf{w}_i^{(\ell)} = -\psi(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t),\ i = \overline{1, n_\ell}\,.$$

The typical form of a finite algorithm is linear

$$\mathbf{w}_i^{(\ell)} = -\Gamma \nabla_{\mathbf{w}} \omega(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t),\ i = \overline{1, n_\ell}, \tag{24}$$

Let MNN training function be as follows:

$$J(\mathbf{x}, \mathbf{x}^{\mathcal{M}}) = 0.5\left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right)^T \left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right).$$

Then

$$\omega(\mathbf{x}; \mathbf{w}_i^{(\ell)}; t) = \left[\mathbf{x}^{\mathcal{M}} - \mathbf{f}(\mathbf{x}, \boldsymbol{\theta}) - \mathbf{g}(\mathbf{x})\mathbf{u}(\mathbf{w}_i^{(\ell)})\right]^T \left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right). \tag{25}$$

In this case equations of adjusting weight factors $\mathbf{w}_i^{(\ell)}$ in MNN layers $\ell = \overline{1, K}$ by virtue of (25) will take the dependence of:

$$\frac{d\mathbf{w}_i^{(\ell)}}{dt} = \Gamma \left(\frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{w}_i^{(\ell)}} \right) \mathbf{g}(\mathbf{x})^T \left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right),\ i = \overline{1, n_\ell}\,, \tag{26}$$

or

$$\mathbf{w}_i^{(\ell)} = \Gamma \left(\frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{w}_i^{(\ell)}} \right) \mathbf{g}(\mathbf{x})^T \left(\mathbf{x}^{\mathcal{M}} - \mathbf{x}\right),\ i = \overline{1, n_\ell}\,, \tag{27}$$

The evaluation of derivatives $\partial\, \mathbf{q}^{(K)} \big/ \partial\, \mathbf{w}_i^{(\ell)}$ in algorithms (26) - (27) is produced in the correspondence with a back propagation error technique:

$$\frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{w}_i^{(\ell)}} = \frac{\partial\, \mathbf{q}_i^{(\ell)}}{\partial\, \mathbf{w}_i^{(\ell)}} \frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{q}_i^{(\ell)}} = \mathbf{f}'(\mathbf{s}_i^{(\ell)}) \mathbf{q}^{(\ell-1)} \frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{q}_i^{(\ell)}};$$

$$\frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{q}_i^{(\ell)}} = \mathbf{W}^{(\ell+1)} \mathbf{f}'(\mathbf{s}^{(\ell+1)}) \frac{\partial\, \mathbf{q}^{(K)}}{\partial\, \mathbf{q}^{(\ell+1)}}.$$

where $\mathbf{s}^{(\ell)} = \mathbf{W}^{(\ell)} \mathbf{q}^{(\ell-1)}$ is first order discriminant function and $\mathbf{f}'(\mathbf{s}^{(\ell)}) = \left. \dfrac{d\mathbf{f}(\mathbf{s})}{d\mathbf{s}} \right|_{\mathbf{s}=\mathbf{s}^{(\ell)}}$ is $(n_\ell \times n_\ell)$ diagonal matrix , $\mathbf{W}^{(\ell)}$ is weight matrix of ℓ th layer.

If the matrix function $\mathbf{g}(\mathbf{x})$ does not depend on a state vector $\mathbf{x}$, i.e. $\mathbf{g}(\mathbf{x}) = \mathbf{B}$, that is taking into account an arbitrary choice of a matrix Γ, substituted to the algorithm (26) - (27) all nonzero elements of a matrix $\mathbf{B}$ can be reduced by unity. It allows to expand the class of parametrical indeterminacy of plant, and assume the matrix $\mathbf{B}$

depending on the vector of unknown parameters $\boldsymbol{\theta}$.

The obtained no autonomous algorithms of on-line network training in a neural net control system structure (26) - (27) are modifications of algorithm SBP. However algorithms (26) - (27) are deprived of the listed above shortages of algorithm SBP and use a measurement information for network training, that adds some adaptive properties in neural net control systems .

Let's illustrate an overall performance of the considered network training algorithms on forced brusselator's model excitation of oscillations example [6]:

$$\begin{cases} \dot{x}_1 = a - (b+1)x_1 + x_1^2 x_2 + u; \\ \dot{x}_2 = bx_1 - x_1^2 x_2, \end{cases}$$

The value of a parameter a is unknown and belongs to area of admissible values $\Omega_{\boldsymbol{\theta}}$, specified by interval $0.1 \le a \le 1$.

We shall choose the MNN parameters as follows: numbers of layers $K = 2$, numbers of neurons in hidden layer $n_1 = 3$, numbers of neurons in output layer $n_2 = 1$.

Based on the control plant equations, algorithms (26) and (27) can be written as follows:

$$\frac{d\mathbf{w}_i^{(\ell)}}{dt} = -\gamma \left(\frac{\partial q^{(K)}}{\partial \mathbf{w}_i^{(\ell)}} \right) \left(x_1 - x_1^{\mathcal{M}} \right), \; i = \overline{1, n_\ell},$$

$$\mathbf{w}_i^{(\ell)} = -\gamma \left(\frac{\partial q^{(K)}}{\partial \mathbf{w}_i^{(\ell)}} \right) \left(x_1 - x_1^{\mathcal{M}} \right), \; i = \overline{1, n_\ell},$$

where $\Gamma = \gamma = 0.1$ is a gain factor.

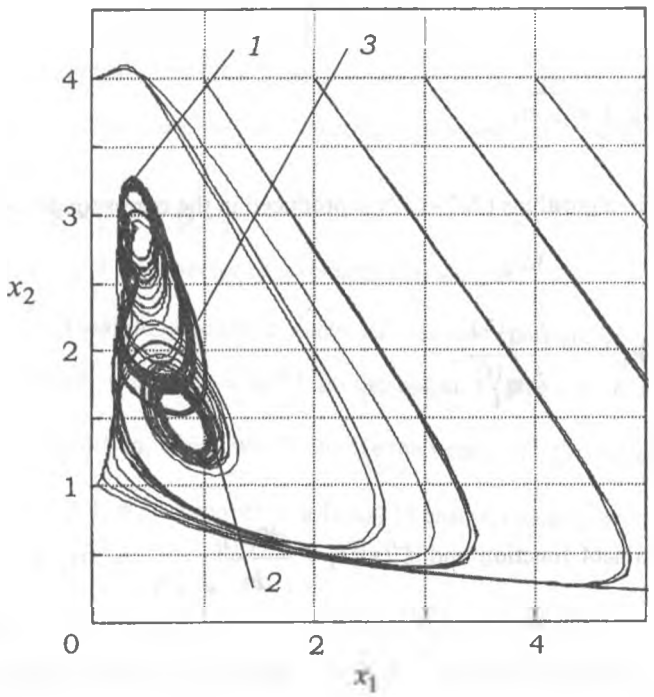


Fig. 3. Phase portraits: *1* - forced brusselator chaotic attractor, *2* - control system with MNN trained on algorithm (26), *3* - control system with MNN trained on algorithm (27)

The following parameters values are chosen for simulation: $a = 0.4 + 0.05\sin(0.81t)$ and $b = 1.2$. At these parameters values in a model phase space there is a chaotic attractor designated as *1* in fig. 3. With the purpose of excitation of oscillations we shall choose reference motion as $x_1^M = 0.8 + 0.5\sin(0.81t)$. The phase portraits of neural net control system, trained on algorithms (26) and (27) are reduced on fig. 3 and are designated as *2* and *3* accordingly.

4. CONCLUSION

In this paper we receive the step estimations for the gradient procedure (3) which provide a training process stability when MNN is included in to the dynamical system (1). Such estimations allow to make a synthesis of an adaptive neural network training algorithms. If use BPTT training technique we can beforehand estimate quality of the neurosystem for the final time interval. It should be emphasize the obtain result may be used in the case when we have a half-final time interval. Future researches will be directed toward an roughness, controlability and observiability of the control system with a neural network.

The introduction of dynamics in multilaered static neural networks training algorithm is added to it properties of a dynamic networks, but without use of feed-backs, as in algorithm BPTT. The process of network training on dynamic algorithm allows to unit in one processes of training and control in static MNN, that is essential at use it as the controller in dynamic systems.

The modifications of speed back propogation algorithm are presented. The efficiency of the proposed algorithms for some chaotic oscillation control problems has been demonstrated. The results of computer simulation demonstrate coincidence of trajectories of own plant motions and trajectories of control system on a significan time interval . It testifies about minimal interference in own plant dynamics and requeres the minimum controls for reaching a control goal.

5. REFERENCES

[1] Terekhov V.A., Efimov D.V., Tyukin I.Y., Antonov V.N. Neural net control systems. — St.Petersburg: Publishing House of St/ Petersburg University, 1999. 265 p.

[2] Terekhov V.A., Yakovlev V.B., Efimov D.V., Tyukin I.Y. The Control and Training Algorithms for Intelligent Systems with Multilayer Neural Networks// The 2-d internat. symposium "INTELS'96" thesis , June 1-4 1996. Vol.1. SPb. - Moscow: 1996. pp 11- 14.

[3] Werbos P.J. Backpropagation Through Time: what it does and how to do it? // Proc. of IEEE. 1990. Vol. 78. pp. 1550-1560.

[4] V.A. Terehov "Dynamic algorithms of multilayered neural networks training in control systems" , Izv. vuz. The theory and control systems, vol. 3, 1996, pp. 70-79.

[5] A.L. Fradkov "Adaptive Control in Complex Systems", Moscow: Nauka, 1990. (Russian).

[6] Fradkov A.L., Pogromsky A.Yu. Introduction to control of oscillations and chaos // World scientific series on nonlinear science / Ed. Leon O. Chua. Ser. A. 1998. Vol. 35. P. 391.

Dynamic Planning in Spacecraft Approaching and Docking Control

Andrew N. Winogradov
Artificial Intelligence Research Center, Program Systems Institute, RAS
Pereslavl-Zalessky, Yaroslavl region, Russia, 152140
phone (+7 08535) 98-993, fax (+7 08535) 20566
E-mail: andrew@andrew.botik.ru

In this paper problems of spacecraft control automatization on the stages of approaching and docking are considered. For solving these problems, we are offer using onboard scheduler, based on rules. Onboard scheduler, based on dynamic planning methods, developing in artificial intelligence researches, is integrated with global knowledge-base, including empirical, experimental and other rules. Also scheduler consider discrete and continuous components interaction.

1. Introduction

The problem of driven objects safe approaching, docking or divergence is actual for many branches (motor transport, air and space vehicles (SV)) and concerns to the most difficult problems of motion control, since is connected to increased risk. Now one from the most important problem becomes a spacecraft safe docking with an orbital space station. The process of docking represents a complex from three goals of control, realizing a full set of possible safe outcomes: tethering, station keeping and safe divergence (fig. 1.).

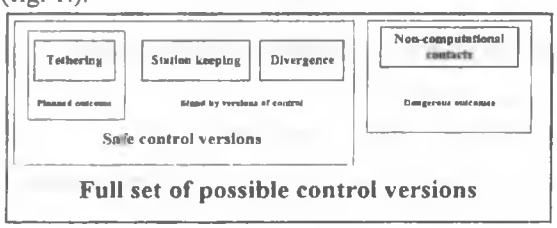


Fig. 1. Possible versions of approach control procedure outcome.

The SV control process can pass in two modes: nominal and abnormal. The nominal mode represents SV control on rigid, beforehand-calculated program. The abnormal mode arises in case of emergencies, such as engine failures, motion control systems failures, failures of radio engineering approaching system or because of time limitations, and assumes transferring the space vehicle control in manual mode. During a spacecraft docking process it is necessary to an astronaut to have a maximum of information about current situation, and also about possible versions of its development. In cause of singularity of docking process, and also in cause of spacecraft design features there are situations, requiring astronaut's immediate operations, that increases probability of incorrect solution acceptance. For the solution of circumscribed above problems is conducted the development of an intellectual onboard space vehicle control system, and within the framework of it the research prototype of the scheduler is offered.

2. Control system model description

The model of a control system actuates the following components:

1) Analytical description of control zones.
2) Database of spacecraft state.
3) Spacecraft model.
4) System of rules.
5) Scheduled trajectory.

Spacecraft model contains:

1) Coordinates of spacecraft mass center.
2) Coordinates and directions of spacecraft engines, relatively to the mass center.

3) Parameters of engines activity.
4) Velocities on axes of coordinates for mass center.
5) Angular velocity vector on axes.

The full set of possible system states is broken on some groups, called as control zones. In each zone there are own control laws, developed on the analysis of long-term statistics of approaching and tethering operations fulfilment, during "SOYUZ" spacecraft manned space flights to orbital space stations "SALUT" and "MIR", and on the research works in the field of hodographs theory [1]. There are chosen the following control zones:

1) Indicator mode zone.
2) Zone of active approaching.
3) Zone of possible tethering.
4) Non-computational contacts zone.

The relative motion of space vehicle is mapped not on a conditional phase portrait, but on a plane of gross error, defined by the vectors of relative distance and relative velocity (Fig. 2.).

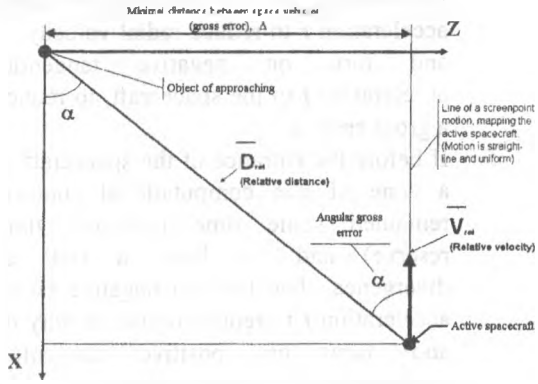


Fig. 2. Mapping of relative motion on a plane of gross error in state coordinates (distance and angle).

Limitations of relation motion control process (Fig. 3.) are represented on a plane of a gross error as a lines, separated the areas:

- Indicator mode, where the control command errors exceed the initial measurement errors, and consequently the approaching control is inexpedient;
- Possible tethering control, where the spacecraft manoeuvrable capabilities provide it docking with given final parameters of relative movement;
- Possible spacecraft station keeping on secure distance from object of approaching;
- The secure divergence of two space vehicles, where the minimum spacing interval between them during manoeuvre fulfilment does not become less than given.

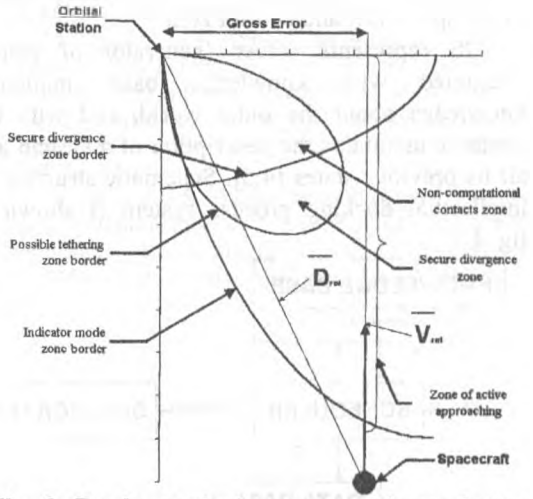


Fig. 3. Basic control areas and boundaries between them.

3. Principles of the intellectual scheduler construction

During a spacecraft docking with orbital station, one of major is the problem of a spacecraft rational trajectory selection. Its solution allows to eliminate (or to minimise) a capability of abnormal situations originating, and also to reduce fuel consumption, that is one of the major factors during a docking process. The offered scheme of the spacecraft mission control organisation allows to select a trajectory of a spacecraft docking at every moment of time, on the basis of the available information about a current position, motion direction and velocity of spacecraft, in view of engines design features, and, depending on control mode, to give: a) the applicable guidelines to an astronaut (manual docking mode), b) the applicable commands on the spacecraft controls.

The kernel of a developed system is the intellectual scheduler (IS), where is realised one of the most effective methods of dynamic processes control, based on the individual plan generation (correction of previous plan) for a current system state in each moment of time (dynamic scheduling) [2,6]. Thus, there is arising a problem of building plans for behaviour control of active object, depending on stages and current goals. Plan – is a sequence of commands (operators) with computed values of parameters. These commands can be described by setting preconditions (conditions of precedent), list of attachments and list of

omissions. Conditions of precedent - formula, which realisability one depends on a current state of the spacecraft and control zone.

IS represents solver (generator of plans), integrated with knowledge base, maintains knowledge about the outer world, and with the database maintains the description of a system and all its previous states [4,5]. Schematic structure of intellectual docking process system is shown at fig. 4.

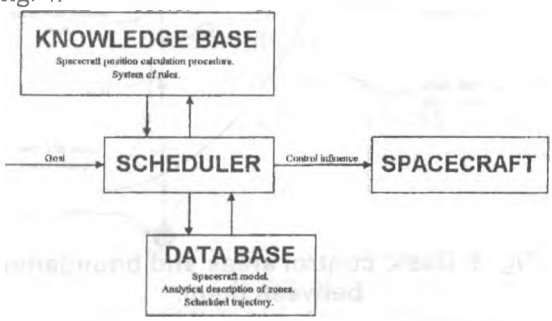


Fig. 4. Intellectual docking process system schematic structure.

The common algorithm of the scheduler includes following actions:

1. Check the target state reaching.
2. Get the necessary information about current system state from the database.
3. Get rules, applicable on existing facts, from the knowledge base.
4. Applying of the rules extracted from a knowledge base, thus updating information about current system state.
5. Check knowledge base for availability of applicable rules, if there are no applicable rules, then go to 6, else go to 3.
6. Build new set of sub-goals on the base of current system state analysis.
7. Rebuild the plan, corresponding to the new set of sub-goals, by adding new and deleting of unrealisable rules.
8. Send the plan on fulfilment.
9. Go to 1.

In given algorithm the methods of deliberative scheduling (dynamic scheduling with time limits)[2,3,6] are used.

System of rules depends on control zones (see fig. 3) and control tasks (tethering, station keeping, divergence). Zones arrangement varies dynamically, i.e. supposed, that each moment of time the spacecraft position is calculated on the base of its relational velocity, using special algorithms [2]. Procedure of spacecraft position calculation works in knowledge base and the outcomes stored in database. Let see the fragment of system of rules of control system (scheduler):

1. If the spacecraft is in a zone of indicator mode, then control is not present.
2. If before the entrance of the spacecraft in a zone of active approaching remained some time (minimal time reserve), then turn on negative tangential acceleration t to the spacecraft, to reduce a gross error Δ.
3. If before the entrance of the spacecraft in a zone of a safe divergence remained some time (minimal time reserve), then turn on positive tangential acceleration t to the spacecraft, to increase a gross error Δ.
4. If before the entrance of the spacecraft in a zone of non computational contacts remained some time (minimal time reserve) and we have a task of approaching, then turn on negative radial acceleration r to reduce radial velocity ω_r and turn on negative tangential acceleration t to the spacecraft, to reduce a gross error Δ.
5. If before the entrance of the spacecraft in a zone of non computational contacts remained some time (minimal time reserve) and we have a task of divergence, then turn on negative radial acceleration r to reduce radial velocity ω_r and turn on positive tangential acceleration t to the spacecraft, to increase a gross error Δ.
6. If before reaching the boundary of station keeping some time (minimal time reserve) is remained, then turn on negative radial acceleration r.
7. If relative velocity V_{rel} is negative, then turn on positive radial acceleration r.

4. Conclusion

Realisation of the intellectual scheduler for approaching and docking processes of space vehicles management will allow to decide many problems of safety, arising during space flights and will reduce number of emergency situations. The demonstration prototype of a system, using concept of motion objects intellectual control, which can be used in various areas of practical application, is now developed.

5. References

1. Бурдаев М.Н. Исследование факторов, определяющих качество моделирования внешней визуальной обстановки на космических тренажерах.// Итоговый отчет № 18-96 по спец. теме. - Российский государственный научно-исследовательский испытательный центр подготовки космонавтов имени Ю.А.Гагарина, 1996.

2. Осипов Г.С. Динамика в системах, основанных на знаниях. // Изв. РАН. Теория и системы управления, 1998, №5, с. 24-28.

3. Фридман И.Г. Алгоритм построения планов в случае взаимодействия целей. III Конференция по искусственному интеллекту КИИ - 92, том 2, с. 104-106.

4. McDermott Drew, Hendler James, Planning: What it is, What it could be, An introduction to the Special Issue on Planning and Scheduling, Artificial Intelligence (1995) 1-16.

5. Osipov. G.S. Applied semiotics and intelligent control.// Proceedings of the Seventh International Conference and Information control systems of robots // Second workshop on applied semiotics.AIICSR'97 (September,15, 1997)- Smolenice Castle, Slovakia, 1997, pp. 27-34.

6. Osipov G.S. Dynamics in integrated knowledge based Systems.// Proc. of the 1998.// IEEE International Symposium on Intelligent Control (September,14-17,1998). - Gaithesburg, Maryland, USA, 1998, pp.199-203.

Algorithms of obstacle detection for autonomous mobile robot control

R. Kh. Sadykhov, A. N. Klimovich, O. G. Malenko.

Belarusian State University of Informatics and Radioelectronics, P.Brovka str. 6, Minsk, Belarus, 220600. Fax/tel(017)2310982. kan@cit.org.by

Abstract

The approach of obstacle detection in this paper is considered. This approach is based on two sequential snapshot analysis. The methods of corresponding points finding are considered. We are using edge detection algorithms for this method approbation.

1. Introduction

The development of the algorithms for mobile robot control is the actual problem. The basic task for such robots consists in achievement of the given point in environment with optimal route, avoiding collision with the obstacles.

The main problem is learning of robot to be guided in some premise (laboratory, room), that have firm flat horizontal floor. The robot is equipped via system of ultrasonic sensors located on perimeter and the infrared scanner, that give a picture in a plane at height 50 sm from the floor surface in a range 180° and radius 2m from center of the robot. As was mentioned, such system of sensors provides good control of the robot in the event that the obstacles on height have been got in a plane of the robot review, otherwise it is necessary to use additional sources of the information about an environment for collision prevention. Such source of the information can be served by a videocamera established on the robot. It gives the junction-type image of environmental conditions on the basis of which it will be possible to determine the form, sizes and situation of obstacles that are not situated in sight of sensors.

2. Methods of obstacles search, based on the videoimage

It is necessary to extract two approaches for task decision of obstacle allocation. First of all one of them can be considered as passive.

Above-mentioned approaches are based on study of one instant snapshot from the camera, extraction and recognition on this image of suspicious zones. In a basis of this approach, as a rule, the rather worked technique of the scene is used [4, 5]. Use of such passive methods has the advantages. Besides, as passive it is possible to consider methods connected to recognition of some key information. As example the special figure on a floor assisting to distinguish it from obstacles, or lines specifying allowable routes, can be served. All this should be used, when there is an opportunity beforehand to facilitate a task of the robot orientation in a room. However it is obvious, that the passive methods are not universal and in a general case with their help it is impossible to receive complete spatial representation of the room. Therefore active methods using idea of stereo-vision are often applied. A number of the authors have considered an opportunity of distance up definition to obstacles, using two cameras, that can be carried in space. This allows to complete three-dimensional representation about an observable scene to be received by processing two images obtained from two points in the space.

The systems on the basis of the image processing via one videocamera were already submitted by some foreign scientists and certain decisions for some special cases are already received. The difficulty of processing is that on the basis of a flat picture of an environment it is impossible to receive volumetric representation about terrain. Therefore we have offered idea to use a movement of the robot for reception of the image of space in front of the robot from two points. This idea is based on simple supervision, according to which person much easier determines distance up to a subject, if the subject is moved (comes nearer or leaves) in a field of sight. That is the representation about a scene can be received by comparative processing of two consecutive snapshots received by movement

of the robot. It is necessary to trace the moving of details on the second snapshot concerning first, and thus to determine a location of the objects in three-dimensional space. The problem is divided into 2 basic subtasks. The first subtask is to distinguish some detail from the first snapshot on the second snapshot and to determine displacement. Second one is to determine its coordinates in space via displacement of a detail. On the basis of this information it is possible to make a conclusion about detail belonging to plane floor and, hence, whether it is possible to consider it as a part of an obstacle.

3. Search and tracking of the image details on two snapshots

This task is analogous to a task of the large image fragment search that most similar on the given small fragment. For its decision various spectral and neural networks algorithms can be used [1-3, 6]. The task becomes simpler and it is possible beforehand to predict in what direction the fragment on the second snapshot will be displaced. That is the size of space for search can be considerably limited and then dimension of a task and time of search are reduced.

Besides it is necessary to develop the method for definition of perspective details in the image for which the moving can be easier traced (easily to separate from an environmental background). It is desirable that these details should be edges of an obstacle, that is key points in space, on which it is possible authentically to define the form of obstacles. For their finding the various algorithms for allocation contours, sides of objects extraction, and also heuristic algorithms designed for a specific room with known obstacles can be used. One of the approaches that allow strongly to simplify this problem can be considered as the following. All key points of obstacles in a room are pasted by the use of some labels and therefore we have opportunity to recognize then on the image from the camera. For example, it can be sheet of the given color (for the color camera) or some symbol (for black-and-white camera). The labels are easily allocated on the first snapshot and their displacement on second snapshot is easily traced. Certainly, this approach is not necessary for orientation in a unfamiliar room, where beforehand it is impossible to put labels. However this approach can be very valuable at a debugging stage of a robot.

Next approach is universal. It based on edges detection and key points definition on them. As key points we consider angles on the images, and such unusual edges as circles, ellipses.

First approach is not appropriate to apply, because this system must be working in unfamiliar conditions, but this approach can be considered as variant for known rooms for more precision navigation.

Second approach is more complex, because it is necessary to extract much details from image and it demands complex scene analysis.

Images, which we have got from videocamera have much color information, and it is necessary to reduce this information. Next features from the images: edges, which contain lines and unusual points, should be extracted.

For solving this problem we need to apply some filtration algorithms. We must make this images more simply.

One of approaches to this simplifying is color reduction. For example if we have 256 colors image, we can reduce them to 64 colors without losing significant information.

Next step of image transformation is edges detection procedure. We can use some approaches for edges detection. More known of them are Sobel and Kalman filters, Hadamard transformation.The results of edge detection presented in the fig.4.

As shown in figures it is possible to allocate sets of lines and to trace them from displacement on the following snapshot. We have considered heuristic approaches for this problem solving.

We have carried out some of heuristic algorithms for unusual key points definition on the first snapshot. We must find them on the second snapshot. The main aim of this algorithms is finding of such points or objects, which it is simply to find on next snapshots (ends of lines, angles, intersections). These algorithms are based on various approaches such as masks applying to transformed image.

4. Algorithm for definition of a point coordinates

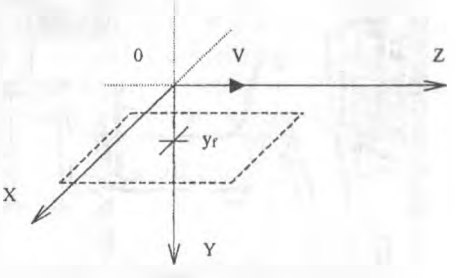


Figure 1 An arrangement of coordinates axes and plane of a floor in a field of the robot sight

We shall consider a case represented in a fig. 1. The robot is forward moved on a floor plane in a V direction, the camera is directed in that direction. Let's arrange axes of coordinates as shown in a figure 1, and we shall establish a point 0 on the center of the camera. The axis Z will be forward directed, and X and Y - to the right and downwards.

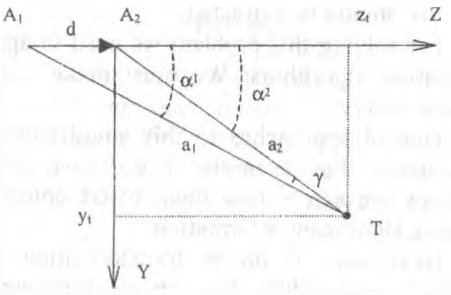


Figure 2. Point coordinates calculation.

Let's robot moved and has passed distance d between points A_1 and A_2, in each of which the image of a room on a course of a movement (fig. 2) was received. Let there is a point Q, that for simplicity are placed directly on a direction of a movement without a deviation on an axis X. That point is allocated both on first, and on the second snapshot. On a deviation of a point from center of the image, knowing the camera resolution, it is possible to determine corners α_1 and α_2 (see fig. 2). Let's determine coordinates of a point O in the moment, when the robot is in a point $A2$.

Then

$$\frac{a_2}{\sin\alpha_1} = \frac{d}{\sin\gamma},$$

where a_2 - distance A_1Q

As $\gamma = \alpha_2 - \alpha_1$, and $z_t = a_2\cos\alpha_2$, that

$$z_t = \frac{d\ sin\alpha_1 \cos\ \alpha_2}{\sin(\alpha_2 - \alpha_1)}.$$

Similarly the following relation can be received:

$$y_t = \frac{d\ sin\alpha_1 \sin\ \alpha_2}{\sin(\alpha_2 - \alpha_1)}$$

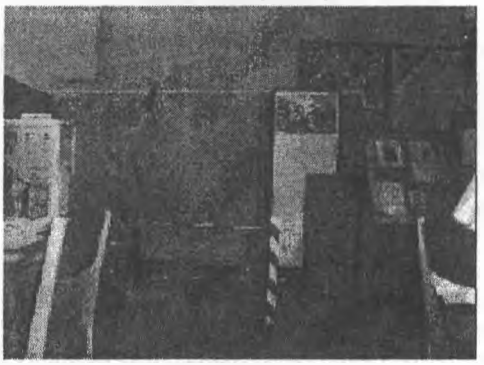


Figure 3. Experimental snapshot

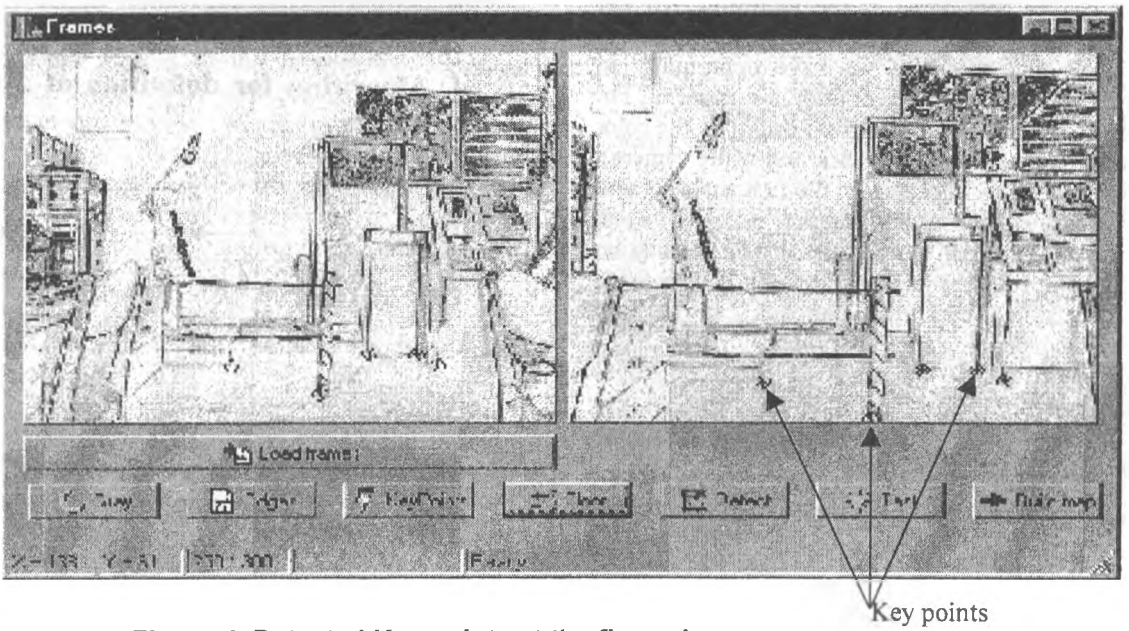


Figure 4. Detected Key points at the floor plane.

These formulas are easily extended for a three-dimensional case, when it is necessary to determine coordinate x_t. Thus it is possible to consider a point in two planes, perpendicular according to axes X and Y. We know vertical and horizontal resolution of the videocamera, then we know angles of reviews on the key point:

$$z_t = \frac{d\ sin\alpha_{x1} \cos\ \alpha_{x2}}{\sin(\alpha_{x2} - \alpha_{x1})}$$

$$y_t = \frac{d\ sin\alpha_{x1} \sin\ \alpha_{x2}}{\sin(\alpha_{y2} - \alpha_{y1})}$$

$$x_t = \frac{d\ sin\alpha_{x1} \sin\ \alpha_{x2}}{\sin(\alpha_{x2} - \alpha_{x1})}$$

On the fig.3 the original videoimage is presented. After image processing we obtain the first snapshot with detected edges. Then we get next image from videocamera and do same transformations. The transformed images in fig.4 are presented. On the basis of z_t and y_t it is possible to make a conclusion about a location of a researched point in space. It should be noted that the most important information is it's belonging in space on an axis Y for a floor plane. The key points which are detected in the floor plane only are shown.

5. Conclusions

The above mentioned algorithms effectively used for reception of a spatial picture of environmental conditions, i.e. the construction of a spatial map of terrain is made possible. However on a way of the decision of this task there are many difficulties, such as shadows, reflections, sharp changes of environmental conditions and etc. The difficulties also arise with moving objects. This algorithm is basically applicable for the static obstacles that are saved their location in space. However given algorithm has also number of advantages, such as:

The optimality of realization from the point of view of hardware expenses, that is the using of one videocamera for formation of a high-grade picture of environmental conditions is possible.

As the image should change on beforehand determined law, the task of definition of the appropriate key points on the image is rather simplified in realization.

Operating with small volumes of the information, that considerably improves stage of the image processing and acceptance of the decision.

Acknowledgement

This work has been performed via project INTAS 97-2028 "Intelligent neural system for autonomous control of a mobile robot". The authors thank the European Community for the financial support of the project. The authors would like to thank Prof. K. Schilling and Prof. H. Roth on granting of experimental base and support our researches.

References

1. A. Held, K.Abe. On the decomposition of binary shapes into meaningful parts. Pattern Recognition, Vol.27, No5, pp.637-647, 1994.

2. Leslie M. Novak, Gregory J. Owirka and Christine M. Netishen. Radar target identification using spatial matched filters. Pattern Recognition, Vol.27, No4, pp.607-617, 1994.

3. D. DeKruger, B.R.Hunt. Image processing and neural networks for recognition of cartographic area features. Pattern Recognition, Vol.27, No4, pp.461-483, 1994.

4. R.Duda, P.Hart. Pattern classification and scene analysis. Stanford Research Institute, Menlo park, California. 1973.

5. F.Martines. Synthesis of the images. Principles, hardware and software: - Moscow.: Radio and communications, 1990. -192p.

6. H.C. Card, G.D. Rosendahl, D.K. McNeill, and R.C. McLeod. Competitive learning algorithms and neurocomputer architecture. IEEE Transaction on computers, vol. 47, no. 8, august 1998.

Artificial Neural Network for DTMF decoders

Andrea Aiello [1], Pasquale Daponte [2], Domenico Grimaldi [1], Linus Michaeli [3]

[1] Dip. di Elettronica, Informatica e Sistemistica, Università della Calabria, 87036 Rende (CS), Italy. Ph.: ++39-0984-494712, Fax: ++39-0984-494713, E-mail: grimaldi@deis.unical.it.

[2] Facoltà di Ingegneria, Università del Sannio, Piazza Roma, 82100 Benevento, Italy. Ph. +39-0824-305817, Fax: +39-0824-21866, E-mail: daponte@unina.it.

[3] Dept. of Electronics and Multimedial Telecommunications, University of Kosice, Letna 9/A, 04120 Kosice, Slovak Republic. E-mail: Linus.Michaeli@tuke.sk.

Abstract

In this paper, the pattern recognition characteristics of the Artificial Neural Networks (ANNs) are used to realise a real decoder for Dual Tone Multi Frequency signals used in the telecommunication field. A new neural architecture, the Multi Learning Vector Quantization (MLVQ) network, is proposed. It offers both greater efficiency in decoding and less sensitivity to noise. In order to solve the problem regarding input signal synchronisation, a pre-processing phase is organised. Respect of the timing parameters required by the international recommendations is assured by implementing a Finite State Machine (FSM). The prototype decoder has been realised by implementing the pre-processing phase, the MLVQ network and the FSM on the TMS320C30 Digital Signal Processor. The decoder has been tested according to the ITU-T Q.24 and Telcordia Recommendations by means of a PC-based automatic measurement station. The test results are given and compared with those obtained by a traditional decoder and by a decoder based on the Multi-layer Perceptron ANN.

Keywords: Artificial Neural Networks, DTMF, Telecommunication, Measurement.

1: Introduction

An increasing number of studies have been carried out on the Artificial Neural Networks (ANNs) in different areas of research. In some problems, they are more efficient than the conventional algorithms and represent an interesting tool for advanced research and applications. In particular, the ANNs promise to be more efficient at recognising and distinguishing complex vectors according to their ability to generalise and form some internal representations of the supplied input signal. These abilities make them very useful for the Dual Tone Multi Frequency (DTMF) signal decoding.

The DTMF signals are commonly used in touch-tone dialling applications [1], home automation via a personal computer [2], interactive banking and reservation systems. They correspond to one of twelve touch-tone digits (0-9, *, #) of the telephone keypad and are the sum of a low frequency tone (typically 697 Hz, 770 Hz, 852 Hz and 941 Hz) and a high frequency tone (typically 1209 Hz, 1336 Hz and 1477 Hz). All the DTMF frequencies have been carefully chosen in order to avoid problems with harmonics and distortion.

DTMF signal decoding is very difficult in real situations. Difficulties arise by a consequence of (i) noise presence, and (ii) frequency and amplitude errors in the generation of the two tones constituting the signals [3, 4]. Others difficulties arise from the international recommendations [5, 6] that establish the parameter values to be respected by real DTMF decoders. These parameters are very restrictive and concern both (i) the signal characteristics, and (ii) the decoding time interval.

Several ANNs could be utilised for DTMF signal decoders. The Learning Vector Quantization (LVQ) network [7] shows the greatest benefits with regard to the characteristics of simplicity, model free classification, and performance.

Nevertheless, the original LVQ network does not permit the realisation of a decoder, which completely satisfies the requirements of the international recommendations.

This paper proposes a solution based on the new Multi Learning Vector Quantization (MLVQ) network. This architecture is characterised by several LVQ networks working in parallel, each one classifying the sampled signal synchronised at a different time instant.

The prototype of the neural-based DTMF decoder has been realised by implementing the MLVQ network on a Digital Signal Processor (DSP). This prototype permits testing of the decoder's performance according to the international recommendations. Respect for the timing parameters is assured by organising and implementing a Finite State Machine (FSM).

All the testing procedures require subjecting the decoder to several DTMF signals with specific characteristics and, consequently, are both complex and very time consuming.

In order to speed up the testing procedures, a computer-based Automatic Measurement Station (AMS) has been designed and realised.

Finally, the results of the tests performed by the AMS are given and discussed. These results are also compared with those obtained by a decoder based on the Multi-layer Perceptron ANN [8, 9] and by a traditional decoder based on a modified Discrete Fourier Transform (DFT) [10]. All the decoders tested are implemented on the same DSP board.

2: The neural DTMF decoder

The block diagram of the neural DTMF decoder is depicted in fig.1.

Before applying the signal to the MLVQ network, the *pre-processing* phase occurs. In the *normalising* block, the sampled DTMF signal is normalised.

Successively, in the *synchronising* block, the synchronising value among the samples is detected in order to synchronise the input signal for the successive classification. Finally, in the *vector organising* block, the appropriate input vector is applied to the MLVQ network.

The *MLVQ* block performs the signal classification. Each block is designed according to the MLVQ network's working characteristics.

In what follows, therefore, the MLVQ network is examined before the pre-processing block.

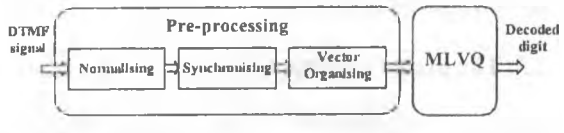


Fig. 1 The Block diagram of the neural decoder.

2.1: The LVQ and the MLVQ network

The MLVQ network is a particular neural structure based on the original LVQ network proposed by T. Kohnen [7]. This last has been modified in order to realise a neural structure able to decode the DTMF signals in the real situations according to the requirements of the international recommendations.

Given N classes of d-dimensional vectors and a vector $x \in \Re^d$, the LVQ neural network classifies x by individualising the class to which it belongs. For this purpose, the ANN needs a training phase, in which a reference vector, called the codebook, is determined for each class. Next, the LVQ network classifies a vector individualising the codebook, which best matches it. The most commonly used matching function (or metric of similarity) is the Euclidean distance defined as:

$$d_e(x,y) = \sum_{j=1}^{n} (x_j - y_j)^2 \qquad (1)$$

The matching function used in this paper is, however, the normalised dot product defined as:

$$d_x(x,y) = \sum_{j=1}^{n} (x_j \cdot y_j) \qquad (2)$$

where the vectors x and y are previously normalised. Experimental tests have shown that the dot product is more accurate than the Euclidean distance in the case under examination.

The input layer of the LVQ network (fig.2) is connected directly to the hidden layer without weighting the signal. The neurons of the hidden layer calculate the matching function between the input vector and their associated codebook according to (2). Each codebook corresponds to a digit (0-9, *, #).

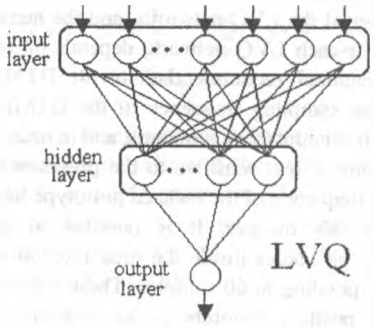


Fig. 2 The LVQ network lay-out.

The output layer neuron calculates the winning matching function as being the largest amongst the output of each hidden layer neuron in order to determine the detected dual-tone. In order to correctly classify the DTMF signals, the LVQ network needs the input vector to be opportunely synchronised. The input vector is applied to the LVQ network in such a way that the synchronising value is the input to a pre-fixed neuron, chosen as reference one.

The absolute peak value is used as the synchronising value. Problems arise from the presence of noise, which causes errors in decoding the absolute peak and consequently an incorrect synchronisation. One solution is to consider more than one synchronising value. In this manner, the DTMF signal is classified by matching the codebooks with the real signal in the neighbour of each synchronising value. Several LVQ networks working in parallel are used. Each LVQ network evaluates the winning matching function (2) for the input vector corresponding to each synchronising value. At the end, the valid decoded DTMF signal is established selecting the largest of the winning matching functions.

The realised decoder is organised by employing the Multi Learning Vector Quantization (MLVQ) network. The MLVQ network consists of a neural structure in which more than one identical LVQ network can be identified. Fig. 3 shows the block diagram of the used MLVQ constituted by four identical LVQ networks working in parallel in order to classify the input vector. The output of each LVQ is the couple constituted by the number associated with the decoded DTMF signal and the value of the winning matching function. The *Selector* block evaluates the largest of the values of the winning matching function belonging to the four LVQ networks. Only if this value exceeds a fixed threshold, is the corresponding DTMF signal assumed to be valid.

2.2: MLVQ network design

The number of synchronising values, corresponding to the number of the LVQ networks, and the number of input neurons for each LVQ network, depends on (i) satisfying the international recommendations for DTMF decoders, and (ii) the sampling frequency of the DTMF signal. By taking into account these problems, and in order to make the decoder robust, less sensitive to the presence of noise, the sampling frequency of the realised prototype has been set to 8kHz. In this manner, it is possible to choose four synchronising values inside the time interval equal to 7.5 ms, corresponding to 60 samples. These values are: (i) the first two positive absolute peaks ordered according to amplitude, and (ii) the first two negative absolute peaks also ordered according to amplitude.

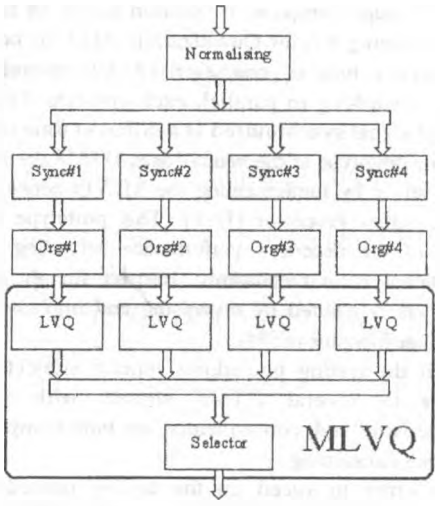


Fig. 3 The flow scheme of the DTMF decoder realised employing the MLVQ.

Several experiments have shown that the MLVQ network with 90 samples for each LVQ network is able (i) to decode the signal in satisfying the international recommendations, and (ii) to minimise the memory required once implemented on the DSP.

The LVQ network is trained in such a way that the input vector includes some elements both preceding and following the synchronising value. The optimal choice, according to the working characteristics of the LVQ network, is to set the reference neuron in the 60^{th} position of the input layer; this neuron's input being the synchronising value. The sampled vector, therefore, is organised, in the *organising* block of fig. 3, so that

- the synchronising value is sent to the 60^{th} neuron;
- the samples before the synchronising value are sent to the neurons preceding the 60^{th};
- the samples after the synchronising value are sent to the neurons following the 60^{th}.

With this organisation of the input vector, the number of neurons of the input layer is set according to the worst case:

1. with synchronising value in the 60^{th} position, the input vector is sent to the neurons from 1 to 90;
2. with synchronising value in first position, the input vector is sent to the neurons from 60 to 149.

In conclusion, the LVQ network with 149 input layer neurons must be used (fig. 4).

2.3: The pre-processing phase

Once the characteristics of the MLVQ network are established, each block of the pre-processing phase can be designed.

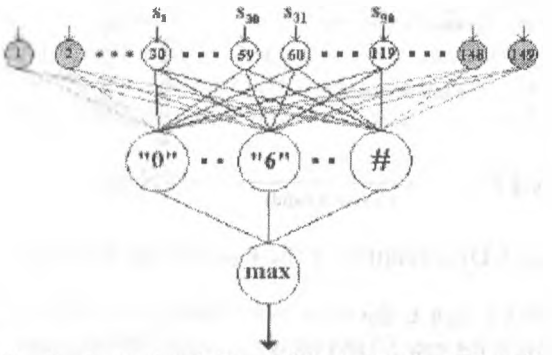


Fig. 4 Layout of the LVQ network with 149 input neurons, 12 hidden neurons, and 1 output neuron. The 60th neuron is the reference neuron.

In the *normalising* block, the signal is sampled and the Euclidean norm of the input vector is evaluated. The normalised vector is then calculated. The output of the *normalising* block is, therefore, the normalised vector of the 90 elements sampled at 8kHz.

The four *synchronising* blocks operate in parallel and in a similar way. The only difference is the synchronising value assumed in each block. In particular: (i) *Synch#1* refers to the absolute maximum peak value in the first 60 elements of the input vector (Fig.5a), (ii) *Synch#2* to the second maximum peak ordered according to amplitude in the first 60 elements of the input vector (Fig.5b), (iii) *Synch#3* to the absolute negative peak value in the first 60 elements of the input vector (Fig.5c), and (iv) S*ynch#4* to the second negative peak ordered according to amplitude in the first 60 elements of the input vector (Fig.5d).

The four *organising* blocks operate in parallel. In particular, (i) *Org#1* provides the corresponding LVQ network with the input vector in which the positive absolute peak value is the input of the reference neuron (Fig.5a), (ii) *Org#2* provides the corresponding LVQ network with the input vector in which the second positive peak value ordered according to amplitude is the input of the reference neuron (Fig.5b), (iii) *Org#3* provides the corresponding LVQ network with the input vector in which the negative absolute peak value is the input of the reference neuron (Fig.5c), and (iv) *Org#4* provides the corresponding LVQ network with the input vector in which the second negative peak value ordered according to amplitude is the input of the reference neuron (Fig.5d). *Org#3 and #4* operate the sign inversion of the samples. In this manner, because the DTMF is a symmetrical signal, the need to train the LVQ#3 and #4 with different training sets is avoided.

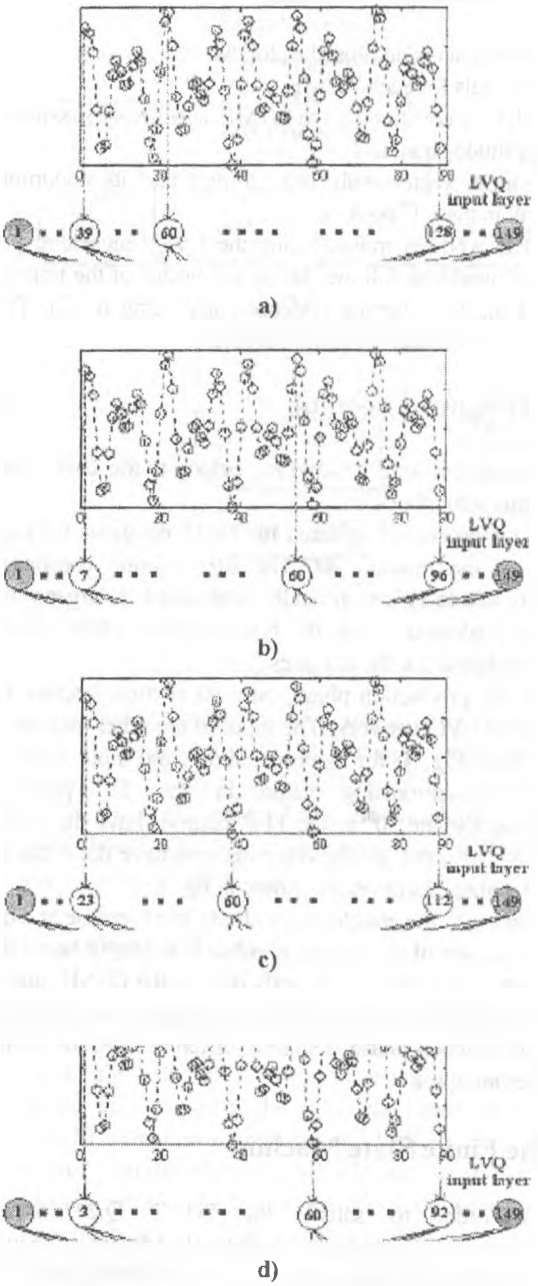


Fig. 5 Synchronising values detected in the first 60 elements and organisation of the input vectors to be sent to the four LVQ networks. The synchronising value, sent to the 60th reference neuron, corresponds to a) the 22nd sample, b) the 54th sample, c) the 38th sample, d) the 58th sample.

3: LVQ network training

The training set is organised as follows:

1. 10 signals for each DTMF;
2. each tone constituting the DTMF signal has a maximum amplitude equal to 1;
3. an input vector with 149 samples and its maximum peak in the 60th position.

The ANN is trained using the LVQ2 algorithm [7]. This last works as follows: let be x a vector of the training set and m_w the winning codebook according to (2). This codebook is updated by using the rule:

$$m_w(t+1)=m_w(t)\pm\alpha(t)[x-m_w(t)] \quad (3)$$

The plus sign is used if x and m_w belong to the same class, the minus sign otherwise.

This process is repeated for 20-25 iterations for each vector of the training set. The $\alpha(t)$ training coefficient (ranging in 0 to 1) is empirically determined. A larger value is usually adopted during the first iterations, while a small one is preferred for the last ones.

In the production phase, only 90 element vectors are sent to the LVQ network. The input of the other neurons is set to zero. Fig. 4, for example, shows the input vector s with a synchronising value in the 31st position. Consequently, the 30th to the 119th neurons have the vector element s_j as input; all the other neurons have the input set to zero. Other examples are shown in fig. 5.

Owing to the architecture's ability to generalise and the optimal choice of the neuron number in the input layer, the LVQ network is able to correctly decode the DTMF signal, opportunely synchronised, even if the input vector utilised in the production phase is slightly different from the vector of the training set.

4: The Finite State Machine

In order to satisfy the ITU-T Q.24 timing specifications, a Finite State Machine (FSM) (fig. 6), which supervises the decoding process, is implemented. In order to guarantee the minimum accepted dual-tone length, the input signal is assumed to have been decoded if in two successive time decoding windows the result of the MLVQ network is the same. Moreover, the FSM allows avoidance of erroneous double-registration of a signal if reception is interrupted by a short break in transmission.

The FSM is controlled by two Boolean variables, *valid and same*, computed in each time decoding window. *Valid* is true if a valid DTMF is detected, while *same* is true if the

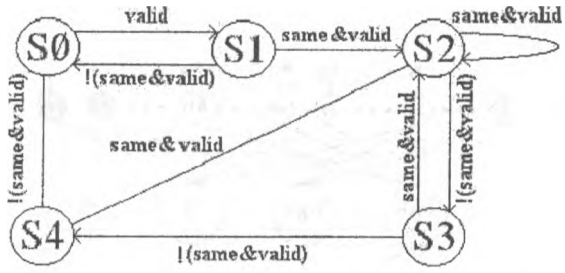


Fig. 6 Organisation of the Finite State Machine.

decoded digit is the same as the previous one. The FSM starts in the state S0 and waits for a valid DTMF signal.

If all the ITU-T Q.24 timing specifications are satisfied, the dual-tone is completely decoded in transition to state S2. From the state S2, the FSM awaits a pause signal before returning to state S0.

5: DTMF decoder implementation on the DSP

Once trained, the pre-processing phase and the MLVQ network are implemented on the DSP TMS320C30 by Texas Instruments [11]. This DSP is a 32-bit floating-point processor, which can execute operations at a performance rate of 33.3 MFLOPS (millions of floating point instructions per second) and 16.7 MIPS (millions of instructions per second). It is installed on an Evaluation Module Board (EVM) equipped with the additional chip TLC32044 used for the sampling section. The maximum sampling frequency of the ADC is 19.2 kHz.

The codebooks are recorded on the DSP memory into an array. The use of the array instead of the matrix makes processing more efficient because only one access is needed instead of the two accesses required by a matrix. The required data memory is 1788 words. In order to efficiently implement the decoding process, a pipeline is realised on the DSP between signal sampling and processing. While the signal is being sampled, the processor analyses the signal sampled in the previous time decoding window. This approach allows the implementation of the FSM and, consequently, the satisfaction of the timing requirements.

6: The automatic measurement station

In order to test the DTMF neural decoder according to all the international recommendations, the Automatic Measurement Station (AMS) has been developed. The AMS uses as its software environment LabVIEW [12] and as its hardware, a PC equipped with a Data Acquisition (DAQ) board made by National Instruments (fig. 7).

It is able both to generate accurate DTMF signals and to acquire the digital signal generated by the decoder, in order to determine the correspondence with the generated DTMF signal. In this way, it is possible to both easily and quickly judge the decoder performance. At the end of each test, a report is furnished in order to summarise the results.

6.1: Hardware design

The DAQ board is the LabPC-1200 [13]. This is a multifunction I/O, compatible ISA board. Its voltage input range is software programmable for 0-10 V (unipolar) or ±5 V (bipolar). The LabPC-1200 has a 12-bit ADC with a maximum analogue signal resolution of 24.4 μV. The single-channel-sampling rate of the ADC is 100 kS/s. This board has two double-buffered 12-bit DACs that are connected to two analogue output channels. Each channel can be independently configured through software for either unipolar (0-10 V) or bipolar (±5 V) operation. The resolution of the 12-bit DAC is 2.44 mV in both cases.

6.2: Software design

The software of the AMS is organised according to the flow scheme of fig. 8. The main blocks are *Test Selection, Test Procedure, I/O Interface and Test Results.*

Test Selection allows the test to be both the ITU-T Q.24 and Telcordia Recommendations. The ITU-T Q.24 Recommendation includes the recommendations of the NTT, AT&T, ETSI and the Australian and Brazilian Administrations. Moreover, the user can fix both the number and characteristics of the test signal in order to guarantee respect of parameters imposed by the recommendations.

Test Procedure is organised to automatically set the characteristics of the test signals according to the previous selected test. This procedure includes the following: *Frequency Deviation Test*, *Power Level Test*, *Twist Test*, and *Timing Test.*

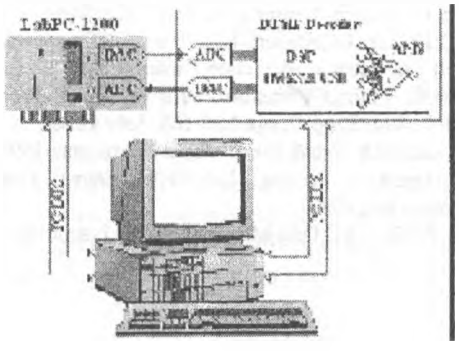


Fig.7 Measurement station for testing the decoder.

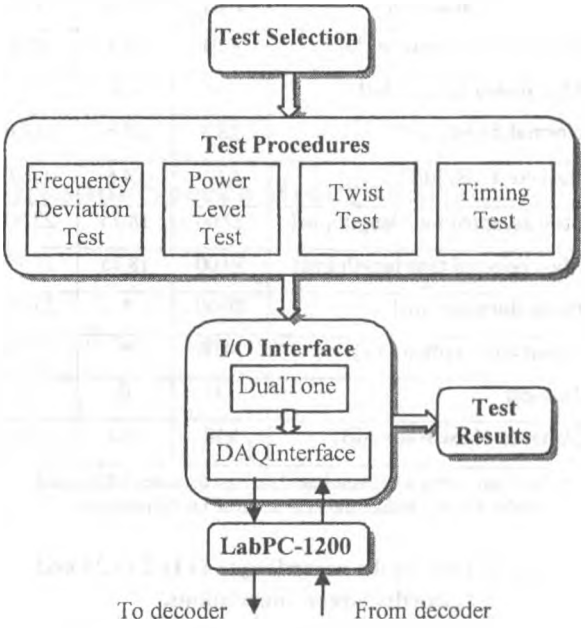


Fig. 8 The flow scheme of the AMS.

The *I/O Interface* includes the two procedures: (i) *DualTone* creates the DTMF signal with assigned characteristics, and (ii) *DAQInterface* administers communication between the AMS and the decoder by using the DAQ board. The *Test Results* generates the final report in order to evaluate the results.

7: Experimental results

In order to test the MLVQ based DTMF decoder implemented on the DSP, the AMS has been mounted as shown in fig.7. In tab.1, the test results are shown and compared with those of the decoders based on a MLP network [8, 9] and on a modified DFT [10].
Compared with the others, the MLVQ decoder shows:

- less sensitivity to frequency distortion of each tone constituting the DTMF signal;
- less sensitivity to the power level of the signal; indeed, it is able to decode signals with lower amplitudes;
- greater capability to decode dual-tone with a difference in power level between the two tones (twist);
- less sensitivity to noise presence on the DTMF signals.

Moreover, the DTMF test tapes have shown that the talk-off performances are good. Indeed, the MLVQ decoder has no false detection when invalid DTMF signals (such as speech) were input. For this test, the Telcordia test tapes were used.

The test results show that the MLVQ decoder meets the ITU-T Q.24 and Telcordia Recommendations.

Parameters	DFT	MLP	MLVQ
Frequency Tolerance [%]	≤2.1	≤1.8	≤1.9
Min. power level [dBm]	-30	-31	-32
Normal Twist [dB]	≤8.3	≤4.6	≤11.0
Reverse Twist [dB]	≤4.3	≤4.6	≤9.0
Min. accepted tone length [ms]	27.00	18.75	22.50
Max. rejected tone length [ms]	39.00	18.75	33.75
Pause duration [ms]	29.00	*	33.75
Signal interruption [ms]	14.00	*	22.50
Talk-off	141	0	0
Signal-to-Noise Ratio [dB]	≥18	≥14	≥12

* These parameters are not evaluated because the tested MLP-based decoder does not include the satisfaction of timing parameter.

Tab. 1 Test results according to ITU-T Q.24 and Telcordia Recommendations.

In particular, differently from the MLP decoder, in the case of the MLVQ decoder the timing respect is assured by the FSM implementation.

In tab. 2, the implementation specifications of the DSP-based decoders are shown. The MLVQ decoder requires a greater data memory than the DFT decoder. However, the memory required by the MLVQ decoder is less than the available memory on the DSP. On the contrary, the MLVQ decoder is characterised by a processing time (sum of the sampling time interval and the algorithm processing time interval) shorter than the processing time of the MLP and DFT decoders, implemented on the same DSP board and using the same sampling frequency of 8kHz.

Other tests were executed in order to evaluate the minimum A/D Converter (ADC) resolution. For the MLVQ based decoder, the minimum ADC resolution is 6 bits, while the MLP-based decoder needs, at least, an 8 bit ADC to correctly decode all the DTMF signals as shown in [8, 9].

Parameters	DFT	MLP	MLVQ
Sampling interval time [ms]	26.60	18.75	11.25
Algorithm processing time [ms]	*	9.00	7.25
Processing time [ms]	26.60	27.75	18.50
Data memory [words]	150	3060	1788

*The DFT algorithm works completely in the time interval between two successive samples

Tab.2 Implementation specifications of the decoders.

8: Conclusions

This paper has proposed the use of a new ANN, the Multi Learning Vector Quantization (MLVQ) network, for DTMF decoders. The MLVQ decoder has been realised and implemented on the Texas Instruments DSP TMS320C30. Due to the intrinsic characteristics of the MLVQ network, the adopted solutions allow the possibility of obtaining a faster and more robust decoder.

The experimental tests have confirmed that the presence of speech and music is no longer a problem. The neural decoder meets the ITU-T Q.24 and Telcordia Recommendations. Moreover, the MLVQ decoder shows low sensibility to noise presence and, consequently, can be used successfully in real applications characterised by a high level of noise.

References

[1] G.Intrater, O.Falik, A single-chip controller for digital answering machines, IEEE Trans. on Customer Electronics, vol. 39 No.1, Feb 1993, pp.45-48

[2] B.Koyuncu, PC remote control of appliances by using telephone lines, IEEE Trans. on Customer Electronics, vol. 41 No.1, Feb 1995, pp.201-209

[3] R.Becker, J. Mulder, SIGFRED: A low power DTMF and signaling frequency detector, IEEE Journal of Solid-State Circuits, vol. 26, No. 7, July 1991, pp. 1027-1037

[4] S.Bagchi, S.K.Mitra, An efficient algorithm for DTMF decoding using the subband NDFT, Proc. of IEEE Symposium on Circuits and Systems, Seattle, WA, USA, 28 April-3 May, 1995, vol. 3, pp.1936-1939

[5] ITU Blue Book, Recommendation Q.24: Multi-Frequency Push-Button Signal Reception, Ginevra,

[6] DTMF Detector Performance http:\\www.ece.utexas.edu \~mason\codesing\DTMF_Performance.htm

[7] T.Kohnen, Self-Organising Maps, 2nd ed., Berlin: Springer-Verlag, 1997

[8] P.Daponte, D.Grimaldi, L.Michaeli, Neural based decoder of DTMF by DSP TMS320C30, Proceedings of the Second European DSP Ed. and Res. Conf., 1998, pp.289-294

[9] P.Daponte, D.Grimaldi, L.Michaeli, Neural network and DSP based decoder for DTMF signals, submitted to IEE Proc. on Science, Measurement and Technology

[10] M.D.Felder, J.C.Mason, B.L.Evans, Efficient dual tone multi frequency detection using the non-uniform discrete Fourier transform, IEEE Signal Processing Letters, vol. 5, no. 7, pp. 160-163, July 1998

[11] TMS320C3X, User's Guide, Texas Instruments, 1994

[12] M.Chugani, A.Samant, LabVIEW Signal Processing, Prentice Hall, 1998

[13] Lab-PC1200/AT, User Manual, National Instruments, 1996

Temporal Processing Neural Networks for Speech Recognition

Alexej V. Ivanov[2], Alexander A. Petrovsky[1]
Computer Engineering Department, Belarusian State
University of Informatics and Radioelectronics
6, P.Brovky Str. 220027 Minsk, Belarus,
tel.: +375-17-2312910, fax.: +375-17-2310914
e-mail: palex@it.org.by[1]
e-mail: alwork@glasnet.ru[2]

Abstract

Application of the temporal processing neural networks (TPNNs) to the speech recognition is justified by the nature of the task. Indeed ASR is a sequence recognition problem and assumes incorporation of time into decision process. Static models treat elements of sequence as independent patterns, which is clearly unrealistic. On the other hand temporal processing nets, built on the basis of multilayer perceptrons give us a hope to dismiss this assumption.

1. Introduction

In our attempt to review the application of the neural networks in the task of speech recognition let us first make use of Marr theory briefly reviewed in [1]. In accordance with it "any complex information-processing system can be studied with respect to three distinct levels of description":

1. *Computational* level – one for description of *the goal* of computation and justification why this goal is appropriate. Here we would try to formulate the task of speech recognition. As we will see later although these formulations are quite diverse they share the common ground of pattern recognition problem, which can be effectively solved with help of neural network approach.

2. *Algorithmic* level – specification of particular algorithms served to achieve the task, specified at the previous level. Here we will restrict ourselves to consideration of temporal processing neural networks (TPNNs) as opposed to static nets, which are already significantly covered in literature. We will present several views onto temporal processing networks, describe their meaning from the signal processing, theory of finite-state automata and probabilistic points. We would try to discuss the drawbacks of chosen approach also.

3. *Implementational* level – specification of the details of realization of the chosen algorithm. Here we will briefly discuss training algorithms developed for temporal processing neural networks.

2. The Speech Recognition Task

2.1 Phonetic Decoding

The earliest attempts to extract information from the speech utterances are at least one-century-old. They had their origin in works of linguists who declared phoneme as the most elementary speech building block. From the

point of view of modern generative phonetics this is not quite true. But let us formulate this task as follows: *Phonetic Decoding – static pattern recognition task with the aim to correctly classify samples coming from one of the limited amount phone classes, phonetic transitions are assumed to take place instantly.*

Practically, there are about sixty different phones distinguished in TIMIT corpus, which constitute the major allophonic realizations of phonemes in the English language. Decoding assumes that each phone has stable target, reached in the process of the generation. There is the problem with stops, their targets are very short periods of time with no airflow and thus no sound, that is why it is a common practice to distinguish two intervals of a stop: closure and release. There are also difficulties with recognition of diphthongs and triphthongs: they have respectively two and three targets successively reached during generation. In any case turning the temporal signal sequence into spatial pattern and manipulating the amount of data fed to the classifier at a time one can find the optimal width of a pattern for phonetic decoding.

2.2 Isolated word recognition

In this formulation our job is to discriminate between speech utterances, which represent isolated words coming from the limited vocabulary. The word boundaries are assumed to be prespecified or found with the help of some external pause detector. There are two ways to extract information from the speech flow:

1. Extract limited amount of features from each utterance, which is in general case is of variable length.
2. Extract limited amount of features per fixed time interval (phone or syllable duration), discriminate between those intervals and generate the word guess with the help of some kind of word temporal structure model e.g. HMM.

There are a number of drawbacks associated with this approach, among the most discouraging we can name:

1. Difficulties with outperforming coarticulation effects at the word boundaries in informal fast speech.
2. Difficulties with introduction of new words to the predefined vocabulary.
3. Impossibility to adequately model word pronunciation variations with the limited amount of features per word.
4. Grammatical forms are treated as different words.

Difficulties with introduction of new words can be outperformed with the help of representation of words as concatenations of syllables, which are modeled at the training stage.

2.3 "Voice control"

This is a task to recognize small amount of simple commands, which are in most cases short sentences built from limited number of words, with some amount of variation of the exact formulation. This task has little difference from isolated word recognition problem, besides the fact that some additional information can be gained from simple grammar. Current state of speech recognition systems allows developing quite reliable "voice control" applications.

2.4 Continuous Speech Dictation Systems

This is the most difficult task of speech recognition. It presupposes usage of well-developed grammar and taking advantage of semantic context. Here we set up a *goal to correctly recognize an arbitrary "well-formed" sentence.* Current continuous speech recognition systems produce a word sequence, which best fits to the perceived utterance as an output. The basic disparity of such formulation with the human way to recognize speech in the fact, that humans able to correct possible grammatical errors of the sentence while understanding the overall meaning, in other words "wellformedness" is not a general precondition for correct recognition.

2.5 Meaning Extraction

This task is frequently associated with researches in the field of artificial intelligence, but from the speech recognition point of view meaning extraction task *constitutes the most abstract representation of utterance at the semantic level.*

2.6 Various improvements

There were several attempts to compare human and machine performance in various speech recognition tasks. The results show that while computers outperform humans in simple

phonetic decoding tasks, humans are much more superior in more complicated dictation and meaning extraction tasks. The main advantages of the human way to recognize speech are:

1. Large vocabulary;
2. Robustness to the environmental noise (so called "cocktail party effect").
3. Speaker Independence – The ability to effectively cope with pronunciation variations from speaker to speaker, regional dialects, foreign accents, etc.
4. Independence of speech rate, ranging from quite slow and clear dictation to fast and often not complete informal conversation utterances.

2.7 Tier representation of speech

Most of the modes employed in recognition treat speech as sequence of some elementary events (patterns, which are classified by recognizer) sequentially concatenated to represent entire message. But linguists for a long time already regard the speech as the communication of information represented at the several tiers: articulatory-acoustic, phonological, grammatical, prosodic, and semantic. These tiers tightly interact with each other in both production and perception processes.

Modeling of such interaction would allow significant improvement of the recognition performance compared to the already developed systems. Many research groups at the time focussed their efforts on this problem [2].

3. Usage of Neural Networks in Speech Recognition

3.1 The Basis

As one can see from the previous formulations of the speech recognition task, all of them share the common idea of statistical pattern recognition: *To label incoming patterns with the probability of misclassification being minimal.* From the statistical decisions theory we know that classifier, which posses this property must assign to the incoming pattern X a class C if the value of posterior probability $P(C \mid X)$ is maximum upon all possible classes. For proof see [3].

Following facts lay in the foundation of usage NN in speech recognition:

- Neural networks trained in classification mode happen to estimate such posterior probabilities (see [4] for proof).
- Neural networks can learn, in other words their parameters can be estimated from some training set automatically with the help of some learning algorithm, without explicit construction by the designer.
- NNs are massively parallel structures, and once properly implemented they can perform their computation very fast.

Neural network models form the broad class of semi-parametric models which is laying between two extremes: parametric models and non-parametric models. Semi-parametric models can be viewed as a compromise characterized by making less constrained assumptions about the process to be modeled than parametric approach while having moderate number of free parameters significantly smaller than in non-parametric modeling.

Many NN architectures were tried in the problem of Speech Recognition, among these we can name MLPs [4], RBF [5], TDNN [5], Recurrent Networks [5] and many other.

Static MLPs, their combination with HMM, RBF networks significantly covered in literature [3], [6], [7], [8] (as well as authors [9], [10], [11]) and lay beyond the scope of this paper.

As the drawback of mentioned approaches we can name the fact that neighboring patterns in the sequence are treated as independent, it is a quite unrealistic assumption. We can at least potentially dismiss this assumption by incorporation time into network operation. Further we'll consider only networks, which are built on the basis of multilayer perceptrons.

3.2 Temporal Processing with Neural Networks

Unlike static pattern classification in sequence recognition we understand that input spatial patterns come as a temporal sequence (figure 1), and as a response we receive temporal sequence of network outputs. The relation between these two sequences is defined by the structure of neural network.

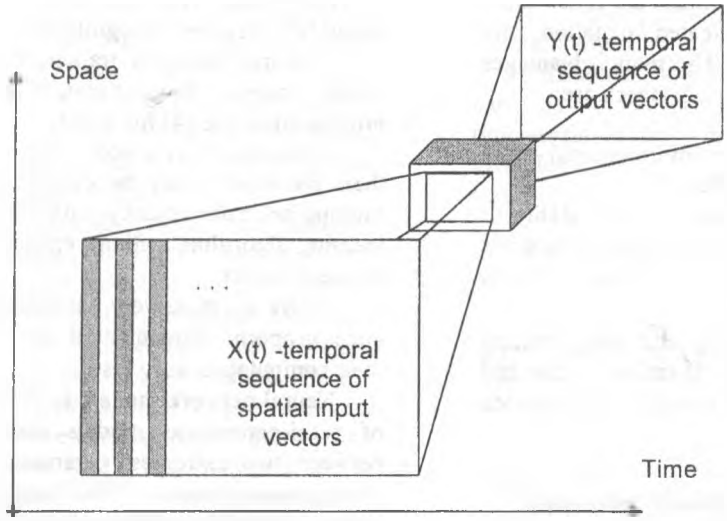


Figure 1. Schematic representation of the temporal processing

There are basically two methods of incorporating time into pattern processing:

1. First way is to present a feed-forward network with some temporal window of input signal (figure 2). Each time the output of such network is computed as a function of input pattern sequence of some finite length:

Thus there is a guarantee that network response for the input pattern sequence of finite length would be also finite.

We have to mention that there are two versions of TLFN: *focussed* and *distributed.* Focussed nets have temporal window only at the input, while each layer of distributed net possesses it's own window, in other words time dependence is distributed through network. Distributed nets can be "unfolded through time" to become focussed at the cost of additional weights.

Universal Myopic Mapping Theorem describes the computational power of the focussed TLFN. It can be stated as follows: *Any shift-invariant myopic dynamic map can be uniformly approximated arbitrary well by a structure consisting of two functional blocks: a bank of linear filters feeding a static neural network.* (After [12])

2. Second way comprises an introduction of recurrent connections between the temporal window of chosen neuron output and neuron inputs of the same or previous level (figure 3). In this situation feed-forward flow is no longer the only direction in which information can be transmitted within a network and each time network output is computed as a function of current input and internal state of the network.

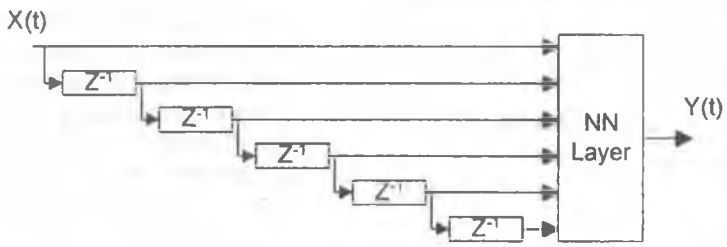


Figure 2. Time Lagged Feed-forward Network (TLFN)

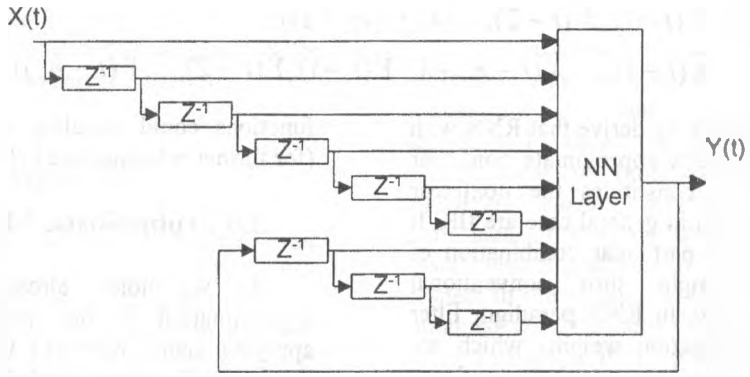


Figure 3. Recurrent Neural Network (NARX model)

Similar to TLFN we can introduce two versions: *globally recurrent* networks only use their outputs as a feedback signal, while in *locally recurrent* networks recurrent connections are introduced at the neuronal level.

Bounded input – bounded output (BIBO) stability criterion is not suitable for RNN, their outputs are always bounded because of neuron saturating output nonlinearity (sigmoid function). This means that RNN's are *always BIBO stable*. That is why discussion of stability RNN as any other dynamical nonlinear system must be done in the Lyapunov sense.

3.3 Delays: Tapped vs. Gamma.

Tapped delay line, characterized by transfer function $G(z) = z^{-1}$ (showed in figures 2 and 3) isn't the only possible way to incorporate short-term memory into network operation.

Let us define the *memory depth* D as a first time moment of total impulse response h_p of the delay line of order p:

$$D = \sum_{t=0}^{\infty} t h_p(t) \qquad (1)$$

Memory depth characterizes ability of a delay line to keep information about the past with a time flow.

Let us further define *memory resolution* R to be a number of taps per unit time interval. Memory resolution defines the quality of the representation of the past.

It can easily be seen that for tapped delay line we have $D = p$, $R = 1$ and their product $DR = p$.

Let us now replace conventional tapped delays with a single pole discrete time filter with a transfer function:

$$G(z) = \frac{\mu}{z - (1 - \mu)}, \quad \text{in which}$$

$0 < \mu < 2$, for the filter to be stable. (2)

This structure would constitute *gamma memory* delay line introduced in [13], it could be shown (see [13], [7]) that in this case $D = p/\mu$, $R = \mu$ and their product remains $DR = p$.

With $\mu = 1$ gamma memory represents ordinary tapped delay line.

With $\mu < 1$ gamma memory is able to store more distant occasions in the past (i.e. increase memory depth) with coarser memory resolution than conventional tapped delay line. The parameter μ can be adapted to achieve the maximum performance.

3.4 Filtering Model.

Here one can clearly see the analogy with FIR and IIR filters. Even more, neural networks of such configurations can be viewed as a generalization of standard filters to the nonlinear filtering. As it was already discussed, the power of neural networks lies in possibility to approximate any finite nonlinear function with arbitrary precision. It should be noted also *that spatial dimensionalities of input and output signals are not restricted* and can be chosen independently one from another.

Thus focussed or distributed TLFN are the approximations of nonlinear non-recurrent filters, which are always FIR, which can be described by the function (3).

$$\bar{Y}(t) = f(\bar{X}(t), \bar{X}(t-1), \bar{X}(t-2), ..., \bar{X}(t-n_x+1)); \quad (3)$$

$$\bar{Y}(t) = g(\bar{X}(t), \bar{X}(t-1), ..., \bar{X}(t-n_x+1), \bar{Y}(t-1), \bar{Y}(t-2), ..., \bar{Y}(t-n_y)) \quad (4)$$

Going further we may derive that RNN with global or local feedback approximate nonlinear function (4), which constitutes the nonlinear recurrent filters, which in general case are IIR. It should be noted that particular combination of the coefficients might turn conventional recurrent filter to FIR. In RNN paradigm filter coefficients are connection weights which are not fixed during training by any relation and it is more convenient to think of RNN as "IIR filter in general".

3.5 State-Space Model

Dynamically driven recurrent networks may be viewed as some sort of finite-state automata. In this case vector $\bar{S}(t)$ is the *state vector* of the model, i.e. a number of internal variables used to store information about past behavior of the model, needed in combination with external input to fully describe its future behavior.

$$\bar{S}(t+1) = f_1(W_i \bar{X}(t) + W_s \bar{S}(t)) \quad (5)$$

$$\bar{Z}(t) = f_2(W_o \bar{S}(t)) \quad (6)$$

in which $\bar{X}(t)$ - external input to the model at a time t, $\bar{Z}(t)$ - model output, W_i, W_s, W_o - connection weight matrixes for inputs, feedback state vector and output respectively. Here we treat multiply delayed output vector $\bar{Y}$ (figure 3) as single state vector $\bar{S}(t)$ and output vector $\bar{Z}(t)$ is computed by as single feed-forward layer with activation function f_2.

If we would brake feedback connections we will have a simple feed-forward MLP computing function f_1 of its inputs, which is capable to approximate any finite nonlinear function with arbitrary precision. Thus our original system can approximate wide class of nonlinear dynamical systems. It should be noted that this approximation holds for compact subsets of input space and finite time intervals.

We also should mention that fully connected recurrent networks with sigmoid activation functions could simulate any Turing machine (for further reference see [7])

3.6 Probabilistic Meaning

As we noted already that probability approximation is our major concern while applying neural networks to the ASR task we have to define the probabilistic meaning of the recurrent networks.

TLFN trained in classification task would approximate the posterior probability of visiting state C at a time t in the form of (7), where $\{\bar{X}\}$ refers the full input sequence, $\{C\}^{-1}$ - state sequence visited at the previous time steps.

It is also possible to shift input window of the focussed TLFN to equally represent future and past contexts. This was done in the NetTalk experiments [14], in the experiments of Morgan and Bourlard [4]; authors also tried such configuration [10]. Such system is not casual (output value depends upon some future values of input) or we can say that there is some delay between the moment the input first time appears in the processing and the time it is associated with some particular class. But human hearing system also posses this property. It is proved in the experiments with temporal masking [15] that during periods of 20-50 ms *before* and 100-200ms *after* loud masker sound faint test sounds perceived attenuated.

Recurrent neural networks trained in the same task will approximate posterior probability as (8), in which $\hat{C}(t-1)$ - represents a network estimate of the state visited at time $t-1$, which might not coincide with the true one.

The problem of replacement of true state with expected one gives rise of a technique called "teacher forcing", which can be described as substitution during the training stage of the possibly wrong network estimate with desired state obtained from the training set. This accelerates training because at some time network may have correct weights, but occasionally be at the wrong place at the state space. On the other hand "teacher forcing"

$$P(C \mid \{\bar{X}\}, \{C\}^{-1}) = P(C \mid X(t), X(t-1), X(t-2), ..., X(t-n+1)), \quad (7)$$

$$P(C \mid \{\bar{X}\}, \{C\}^{-1}) = P(C \mid X(t), X(t-1), ..., X(t-n_x+1), \hat{C}(t-1), ..., \hat{C}(t-n_y)) \quad (8)$$

removes all information about previous errors made by network. Globally this would lead to optimizing error function different from "unforced" case.

3.7 Training algorithms

A number of training algorithms was developed for TLFNs and recurrent MLPs on the basis of standard backpropagation algorithm.

Focused TLFNs can be trained with standard backpropagation algorithm converting temporal input signal into spatial input vector. This procedure can be done once at the stage of preparing training or testing sets.

Any distributed TLFN can be "unfolded" in time to become equivalent focussed network with much bigger input window size and some amount of shared weights and trained with "static" version of backpropagation, but this procedure seems to be impractical. A special procedure called *Temporal Back-Propagation* algorithm was proposed by Wan [16] for the case of distributed TLFN.

Now, that we turn to the brief discussion of the training algorithms for the recurrent networks let us first define two properties:

Algorithm is thought to be *local in time* if it can be executed as temporal input sequence arrives and allows learning using only information contained in the temporally neighboring frames of input signal. Such algorithm can be used to learn sequences of arbitrary length.

Algorithm is *local in space* if weight updates of each neuron can be computed only form the information about immediate neighbors of chosen neuron. Such algorithms can be easily implemented in parallel mode.

For the globally recurrent network locality in space and time are alternatives and can't be achieved simultaneously without any simplifying assumptions.

Back-Propagation Through Time was proposed in [17] as recurrent extension of the standard algorithm and can be summarized as follows:

1. Forward propagation of the input sequence of fixed length, memorizing each neuron's activation at every time step.
2. Backward computation through space and time of correction values for each weight in the network.

This algorithm resembles backpropagation through equivalent "unfolded" feed-forward network. This "unfolding" procedure is possible because input sequence is restricted to limited length. Even more a truncated version of BPTT algorithm was introduced [18], which rejects longer time dependencies than some predefined length. BPTT algorithm is local in space.

A local in time alternative (*Real-Time Recurrent Learning*) was proposed in [19]. The core idea of this algorithm is in the use of instantaneous gradients of the cost function with respect to the weights in the network. Gradients obtained with the help of RTRL would deviate around values of BPTT gradients. This deviation is exactly analogous to the behavior of on-line update technique in front of batch update in conventional static backpropagation.

Further comparison of BPTT and RTRL reveals that while BPTT is computationally simpler and more effective than RTRL, RTRL is casual and therefore suitable for continuous learning without explicitly predefined training set.

Recently several other training algorithms (Recursive Backpropagation, Casual Recursive Backpropagation) were introduced for the case of recurrent networks with local feedback [20]. The main advantage of such algorithms is the fact that training algorithm can be simultaneously local in time and local in space.

3.8 Universal Time flow Rate

But in spite of generality of presented models, one important limitation can be noted in these formulations. All models mentioned above have a property of *universal time flow rate*, i.e. one time step at the input strictly cause exactly one time step at the output. Besides the fact that this brings extra computational complexity (all parts of the network should operate at the time scale of the input signal), it is closely related to the "vanishing gradients" problem (for complete description [5]), in short, recurrent network fails to memorize long term dependencies.

To illustrate time scale problem, let us consider the problem of phoneme identification. If we build a classifier based on TLFN or RNN (for particular examples see [4], [5], [10]) we are forcing the network to produce its outputs at a rate of input preprocessed acoustic signal, which can be considered as a piecewise stationary process through a time period of measurement, but anyway at the phoneme boundaries speaker articulators are in transition to the next stable target configuration. Additional models should be used (such as tri-state phone HMM) to generate global decisions about produced

utterance. In spite of the fact that in classification mode the network approximates posterior probabilities on the basis of which the optimal classification should be done, it does this "too fast".

3.9 Why temporal processing neural networks did not solve speech recognition problem

Feed-forward and recurrent MLPs possess many useful properties for speech recognition, these models have well understood signal-processing state-space and probabilistic meaning, they naturally adopt training, that is why they can't be easily rejected as candidate models for speech recognition. But from this point we already can see some drawbacks of this approach comparing to the problem to be solved. Let us briefly review the basic disparities between the general speech recognition task and recurrent network made of sigmoid neurons:

1. "Linguistic tiers" representation of speech assumes sequential decoding at various levels (i.e. transformation of acoustic signal to phones, further transformation from phone sequences to syllables, syllable sequences to words, word sequences to sentences and so on up to the meaning.). Linguists insist (with some evidences) on non-feed-forward information flow between such tiers (so called "tier interaction"). Moreover, time flow rate is *slowing down* in the direction of higher abstraction levels; in other words one phone constitutes some sequence of acoustic vectors. On the contrary, TPNN provide us with uniform time flow rate mapping between input and output sequences. This problem was addressed from the various positions (Dynamic Time Warping, statistical models like Hidden Markov Models, e.t.c.) but all this approaches suffered from the difficulties of introduction of new higher abstraction entities, such as problems with incorporating new words into a limited dictionary speech recognition system.

2. The size of output alphabet grows very fast from tier to tier from less then hundred different phones to several thousand syllables (There are about 8000 distinct syllabic structures in English language in accordance with [2]), tens of thousands words, virtually unlimited number of shades of meaning. It clearly becomes impractical to encode each possible output class with separated neuron of the output level of the network at the higher processing stages.

3. Learning algorithms developed for MLPs assume more or less equal representation of samples coming from different classes at the training stage in order to achieve equal modeling power for the various output classes. This requirement is clearly unrealistic in continuous real-time learning procedure.

The majority of the problems could possibly be overcome with the help of Pulse-Coupled Neural Networks (PCNN) (for comprehensive foundation look [21], [22]), an approach with biological grounds frequently mentioned as more precise model of biological neuron introduced by Reinhard Eckhorn in 1990. Despite the fact that works of Eckhorm were inspired by specific activity in *visual cortex* of small mammals and the most of known applications of PCNN are in the field of image processing PCNNs posses several useful features for speech processing also:

1. Output neuronal activity represents series of short spikes with a rate proportional to a sigmoid function of the feeding input.

2. Synchronous groups of neurons act as "bigger neurons", operating at the slower time scale, firing not a single spike at a time, but series of spikes with different amplitude.

3. Signatures of that spike series can be viewed as a way to encode output classes, which gives coding capacity bounded by the total number of neurons (not only in the output layer), *coding capacity grows with time scale slowing down*. This is exactly what we have observed for higher speech processing stages.

4. Signature output coding gives an opportunity to introduce a kind of distance measure between different output classes as opposed to "one from m" coding, where there is no possibility to gain any similarity measure from class labels themselves.

Unfortunately learning algorithms for PCNN are not well understood and developed yet, but the principal possibility of such algorithms exists.

4. Conclusion

Temporal processing neural networks provide more suitable framework for speech recognition problem comparing to the conventional static MLPs. They use short-term context information in more effective way than their static counterparts. They provide us with the models, which have less free parameters to be estimated during training.

In their current state they can replace MLPs in the HMM/MLP hybrid recognition system at the phonemic decoding stage.

But even though they have all these useful properties, TPNNs can't be considered as homogenous devices for solving speech recognition problem in general due to the reasons discussed here.

References.

[1] R. M. Golden "Mathematical Methods for Neural Network Analysis and Design" The MIT press, Cambrige, 1996

[2] S. Greenberg "On The Origins of Speech Intelligibility in the Real World", Proc. Of the ESCA Workshop on Robust Speech Recognition for Unknown Communication Channels, April, 1997

[3] B. Ripley "Pattern Recognition and Neural Networks" Cambridge University Press, 1996.

[4] H. Bourlard, N. Morgan "Connectionist Speech Recognition, A Hybrid Approach" Kluwer Academic Publishers, Boston, Dordrecht, London 1994.-312p.

[5] Y. Bengio "Neural Networks for Speech and Sequence Recognition"// International Thomson Computer Press, 1996.

[6] C. Bishop "Neural Networks For Pattern Recognition" Clarendon Press, Oxford 1995.-482p.

[7] S. Haykin "Neural Networks: A Comprehensive Foundation" Prentice Hall Inc., 1999

[8] F. L. Luo R. Unbenhauen "Applied Neural Networks for Signal Processing" Cambridge University Press, 1998

[9] A. Ivanov, A. Petrovsky "Training Multi-Layer Perceptrons in the problem of Static phoneme Identification with the use of TIMIT Speech Corpus", Proc. Of 6th International Workshop on Systems, Signals and Image Processing, 1999 June 2-4, Bratislava, Slovakia;

[10] A. Ivanov, A. Petrovsky "Experiments with Neural Networks for Sequence Recognition in Application to Automatic Speech Recognition" 5th International Conference on Pattern Recognition and Information Processing, May 18-20, 1999, Minsk, Belarus;

[11] A. Ivanov, A. Petrovsky "MLPs and Mixture Models for the Estimation of the Posterior Probabilities of Class Membership", A Workshop on Text, Speech, Dialog TSD'99 Sept. 13-17, 1999, Plzen, Czech Republic.

[12] I. W. Sandberg L. Xu "Uniform approximation of multidimensional myopic maps", IEEE Trans. on Circuits and Systems, vol.44, pp. 477-485, 1997

[13] B. De Vries, J. C. Principe "A gamma model – A new neural model for temporal processing" Neural Networks, vol.5, pp. 565-576, 1992

[14] T. J. Sejnowski, C.R. Rosenberg "Parallel networks that learn to pronounce English Text", Complex Systems, vol. 1, pp. 145-168, 1987

[15] E. Zwicker, H. Fastl, "Psychoacoustics: facts and models" Berlin: Springer-Verlag, 1990, - 354 p.

[16] E. A. Wan "Temporal backpropagation for FIR neural networks" IEEE Int. Joint Conf. On Neural Networks, vol. 1, pp. 575-580, San Diego, CA, 1990

[17] D. Rumelhart, G. Hinton, R. Williams "Learning Internal Representations by Error Propagation" Parallel Distributed Processing, v. 1, ch. 8, pp. 318-362, MIT Press, Cambridge, 1986

[18] R. Williams J. Peng "An efficient gradient-based algorithm for on-line training of recurrent network trajectories" Neural Computation, vol. 2, pp. 490-501, 1990

[19] R. Williams, D. Zipser, "A learning algorithm for continually running fully recurrent neural networks" Neural Computation, vol. 1, pp. 270-280, 1989

[20] P. Campolucci, A. Uncini, F. Piazza, B.D. Rao "On-line Learning Algorithms for Locally Recurrent Neural Networks" IEEE Trans. on Neural Networks, vol. 10, n. 2, March 1999

[21] O. Omidvar, J. Dayhoff (ed.) "Neural Networks and Pattern Recognition" Academic Press, 1998

[22] Special Issue on Pulse Coupled Neural Networks, IEEE Trans. on Neural Networks, vol. 10, n. 3, May 1999

Intelligent System for Prediction of Sensor Drift

V. Golovko[1], J. Savitsky[1], A. Sachenko[2], V. Kochan[2], V. Turchenko[2],
T. Laopoulos[3], L. Grandinetti[4]

[1] Brest Polytechnic Institute, Belarus, cm@brpi.belpak.brest.by
[2] Ternopil Academy of National Economy, Ukraine, sachenko@cit.tane.ternopil.ua
[3] Aristotle University of Thessaloniki, Greece, laopoulos@physics.auth.gr
[4] University of Calabria, Italy, lugran@unical.it

Abstract

In this paper the features of neural networks using for improve of measurement accuracy of physical quantities by sensor drift prediction are considered. There is use a technique of data volume increasing for training of predicting neural network by using of separate approximating neural network.

1: Introduction

The new technologies require the greater quantity of the measuring information that has resulted recently in significant development of its reception means. High accuracy and significant information processing possibilities characterize the modern measuring systems. However in majority of cases is rationed (installed) the measurement error of output sensor signal instead of physical quantity. In the last decades the accuracy of sensor signal measurement has increased in dozens of times. However the analysis of [1] and [2] has shown that the sensor error has insignificantly decreased for this time. For example, at temperature measurement by Honeywell Pt100 sensors [1] and block Hydra 2625A Fluke [3] a ratio of errors of measuring channel elements more than fifty. The development of computing means has allowed considerably to increase a degree of information processing (to use complex mathematical methods of processing and to operate with knowledge of measurement object). But majority of work, which devoted to sensor signals processes [4, 5, 6, 7], consider questions that not connected with improving of measurement accuracy of physical quantity at measuring systems exploitation. In the works [8,9] using of artificial intelligence methods for improving of measurement accuracy of physical quantities are discussed.

In this work the methods predicting of sensor drift by artificial neural network system are considered.

2: General Structure

Intuitively considering intelligence in signal processing systems, the existence of a central computational processing unit is obvious. The higher the number of signals (here sensor devices), the grater the computational power needed and heavier the load of signal transmission through the system. A preferable structure is of a distributed signal processing system, were sensor device(s) information is being processed in an intermediate level and only "useful" information is transmitted to a higher hierarchical level. Considering the sensor devises as a sensors (actuators) and sensor interface circuits and the need of a human-machine interface, a general structure of a multi-sensory, multi-modal system is presented in Fig. 1. The realization of such structure is feasible under the premise that the computational power hidden in the processing levels is adequate to perform the operations, which will give to the system four basic properties: adaptability, accuracy, reliability and universality.

1. Accuracy

An accurate system must be able to compensate systematic (offset, gain, nonlinearity, cross sensitivities), systematic drift and random errors originated from sensors characteristics or system parameters. The ability of dealing with missed data due to random (transient or intermittent) faults is also desirable.

2. Universality

The universality system must provide following possibilities:

- Application of ISIS for the solving of various problems. It means the ISIS application for measurement of wide number of physical quantities. The modular structure of wide operating hardware and universal software are used for this purpose;
- The possibility of ISIS easy increasing. This ISIS property means the development and wide use of the hardware and software modular libraries. To

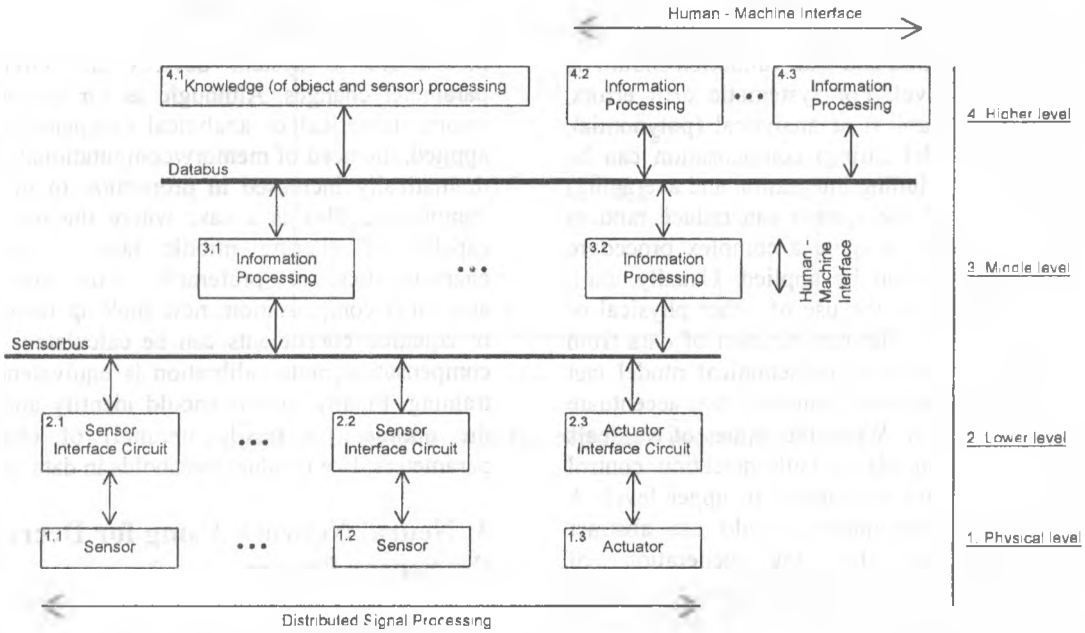


Figure. 1. General Structure of ISIS

add the new interface sensor circuit it is sufficient to have its driver. The operating program of middle level node is compiled at the high level taking into account all necessary drivers, and it is recorded to the node in remote reprogramming mode.

2.1: Distribution of operations in system levels

Having defined the system's main principles, the next step in modeling its general structure is the definition of all possibly required operations and their distribution at the various levels. Following the structure of Fig. 1 bottom-up such a description is made in the rest of the paper.

2.1.1: Physical level.

Physical level includes the sensors/actuators whose inputs are the physical quantities and outputs are signals of voltage, current or time (period or frequency in a form of pulses). In the case of actuators, excitation signals provided by the lower level is needed. The sensors interface circuit (or MM) should provide the connection of different sensors and are classified according to sensors signals, but not according to sensors types.

2.1.2: Lower level.

The sensor interface circuit must receive the sensor's signal and manipulate it to provide a digital output signal comprehensible by the middle level. Possible operations executed here are:

- Amplification.
- Filtering (DC rejection, separation of common mode from differential mode signal etc.).
- Analog to digital conversion;
- Switching.

Also, when many similar sensors (physical redundancy) are used, they can share the same interface circuit or parts of it by multiplexing. Excitation signals to actuators are supplied by the circuitry of this level. So, the wiring required between physical and lower layer is for data and actuator excitation signal transferring. Finally, as to the adaptability rule, a digital to analog conversion of the control signals provided by the higher hierarchical (middle) layer might be required. The lower level must provide digital to analog conversion for creation of sensor activaton signal (for example, set of working current RTD). Also it is sufficient to have some 8-bit DAC, its output voltage can be measured by 16-bit ADC. The universality rule requires the possibility of different types of sensor interface circuit usage in ISIS. They must provide interaction with maximum number of sensors.

2.1.3: Middle level

This layer must carry out three important tasks, control the lower level units, collect and process information from them and communicate with the upper layer. Such performance can be accomplished by the use of either a high performance microcontroller (μC) or a Digital Signal Processor (DSP).

According to adaptability and reliability rules, middle level must control sensor and sensor interface circuit modes and ranges (adjusted to improve sensitivity) as well as the multiplexing (depending on desired data rate) of the various sensor devices to the sensor bus. This leads to a bi-directional sensor bus for digital data and control signals transferring. Since either parallel or serial bus can be applied, the choice is made by compromising between data rate and

wiring reduction. However latest μC and DSP include high-speed synchronous and asynchronous serial ports, making serial bus selection extremely appealing.

Error compensation and data validation should de performed in this level. For systematic drift errors, numerical (look-up tables) or analytical (polynomial, exponential etc. model fitting) compensation can be applied. Filtering (including integration and averaging) and proper design of the system can reduce random errors. Data validation is quite a complex procedure and various methods can be applied. Usually, fault detection is achieved by the use of either physical or analytical redundancy. The combination of data from similar sensors or sensor's mathematical model can generate residuals, analytical functions that accentuate in the presence of fault. When the values of residuals exceed defined thresholds, a fault detection control signal is generated and transmitted to upper level. A more advanced implementation would use abstract computing techniques for the generation of characterized fault detection signals, depending on the fault's source and duration as well as the contribution of fault to unreliability of data.

An other appealing method of error compensation and data validation would be the use of Artificial Neural Networks (ANNs). The drawback of the realization of ANNs in this layer of the system is the computational power required in the training phase. Feasible solution could be the execution of the training phase at the upper layer and the transferring of the computed neuron synapse weights down to the middle layer. Application of ANN metrologies will be further discussed on the upper level session.

Communication between middle and upper layer includes the transferring of data and status signals upwards and data and control signals downwards. Standard protocols like RS232, RS485 or IEEE1451 can be applied. The DataBus should provide the following requirements:

- The network topology is the common bus with the branching possibilities;
- The total network length should be not less than two kilometers;
- The possibility of the additional users connection to the network without its turning off;
- The cheapness cost of cable and network accessories;
- The maximum usage of already existing equipment.

2.1.4: Upper level

The main computational unit of the system collects and combines data and status signals from the various information processing devices and makes the abstract application-depended decisions. Relating to system's adaptability and reliability, self-testing and auto-calibration should be performed here.

Self-testing refers to the intelligent function of monitoring each and every sensor device and detecting of any unreasonable behavior. In case of such detection, attempt of (auto) calibration or isolation of the device while signaling for maintenance or replacement should be performed.

Auto-calibration is the system's adaptation mechanism to system devices and environmental parameter changes. Although, as for systematic drift errors, numerical or analytical compensation can be applied, the need of memory/computational power will dramatically increased in proportion to the system's complexity. This is a case where the use of ANNs capable of altering middle layer's compensation characteristics is preferable. For numerical or analytical compensation, new look-up table elements or equation coefficients can be calculated. For ANN compensation, auto-calibration is equivalent to ANN training. Finally, ANNs should identify and maintain the optimal (or nearly optimal) of characteristic parameters, like residual thresholds in data validation.

3: Neural Network Using for Decreasing of the Sensor Errors

The sensor error is determined by initial spread of its conversion characteristic (at manufacturing) and by its drift in operating conditions [2]. First component is corrected relatively easy. The drift of the majority of the sensors is characterized by complex temporary functions where its parameters depend on operating conditions [10]. Thus the drift prediction after results of preliminary researches of particular type of sensors provides sufficient reliability of accuracy improving only in separate cases. The reliable improving of accuracy irrespective of operating conditions is provided by periodic testing of sensors with standard sensor or by using of special calibrator for sensor's calibration. The highest accuracy is reached at testing or calibration on operating place, but the operations that realize these methods are reasonably laborious. The decrease of costs on their realization is possible by decrease of their fulfillment frequency at the expense of high-quality prediction of sensor drift during inter-testing interval.

It is necessary to note that the functions of sensor drift usually have individual character and have significant casual component [11]. Prediction of such functions is reasonably a complex problem. For its solving it is offered to use neural networks that are optimum for problems of such kind [12]. It is defined by adaptive properties of neural networks at the expense of its self-training. It is known [13] that the quality of neural networks training in a strong degree depends on using data volume. However the aspiration to increase the inter-testing interval at the expense of high-quality prediction of sensor drift proportionally reduces numbers of data for neural network training. The method of artificial increasing of data volume is offered by using of approximating and predicting neural networks. The first network approximated the real data and permits to generate data volume which sufficient for training of predicting neural network. The necessity of two neural networks using is defined by the different requirements to their properties and their internal structure.

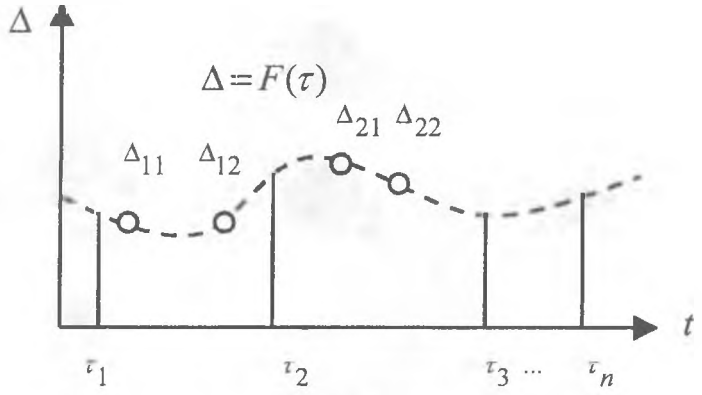


Figure 2. Sensor errors for different moment of time.

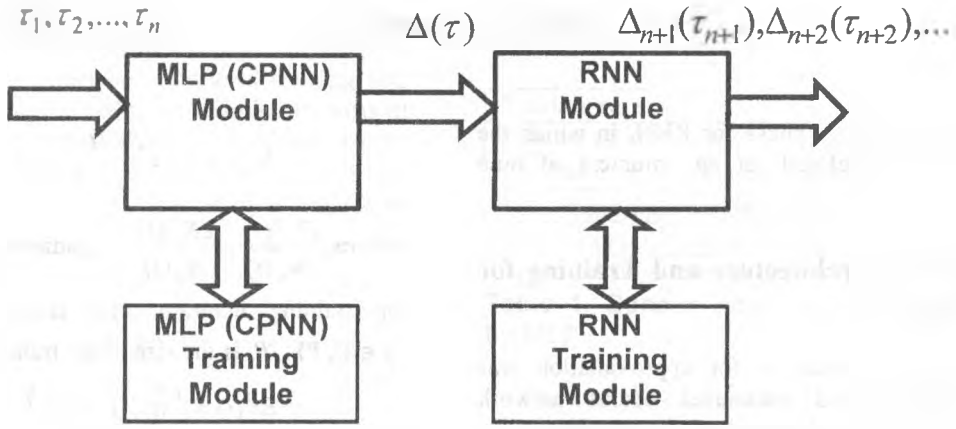


Figure 3. Sensor error prediction scheme.

The objective of the neural system is to predict the sensor error for any moment of time: $\tau > \tau_n$ *and* $\tau \in [\tau_1, \tau_n]$. For achievement this goal is necessary to solve the following tasks:

1. Given a finite sequence $\Delta_1(\tau_1), \Delta_2(\tau_2), ..., \Delta_n(\tau_n)$, find the meaning of error Δ for any moment time $\tau_{ij} \in [\tau_1, \tau_n]$:

$$\tau_{11} \to \Delta_{11}, \tau_{12} \to \Delta_{12}...$$
$$\tau_{21} \to \Delta_{21}, \tau_{22} \to \Delta_{22}$$
$$.....$$
$$\tau_{n-1,1} \to \Delta_{n-1,1}, \tau_{n-1,2} \to \Delta_{n-1,2}$$

2. Given a finite sequence for any moment of time $\tau \in [\tau_1, \tau_n]$, find the continuation the time series $\Delta_{n+1}(\tau_{n+1}), \Delta_{n+2}(\tau_{n+2}), ...$

Such approach permits to predict the sensor errors for any moment of time.

The common architecture of the neural system is presented on Fig. 3. It consists of two neural modules. Multilayer perceptron (MLP) or counter propagation neural network (CPNN) can use as the first module. It is meant for the approximation of the function $\Delta = f(\tau)$. In result is obtained the training set for the second module. The recurrent neural network is used as the second module. It is meant for predicting sensor errors.

The use of this neural system involves several separate phase, which have to be follows:

1. The learning of MLP or CPNN, in which suitable training set is used to train the neural networks. As far as the training algorithm is concerned, the backpropagation or counterpropagation algorithms can be applied. A more detail information about application of this algorithms will be gave below.
2. The validation phase for MLP (CPNN) [8]. In this case the neural network's generalization ability is verified by means of other data. Also the neural network accuracy is estimated.
3. The production phase for MLP (CPNN). In result can be obtained the training set for the recurrent neural network.
4. The learning of RNN. The previously obtained data are used for training of the RNN. As far as the learning algorithms concerned, it will be gave below.
5. The validation phase for RNN, which is performed same as previously.

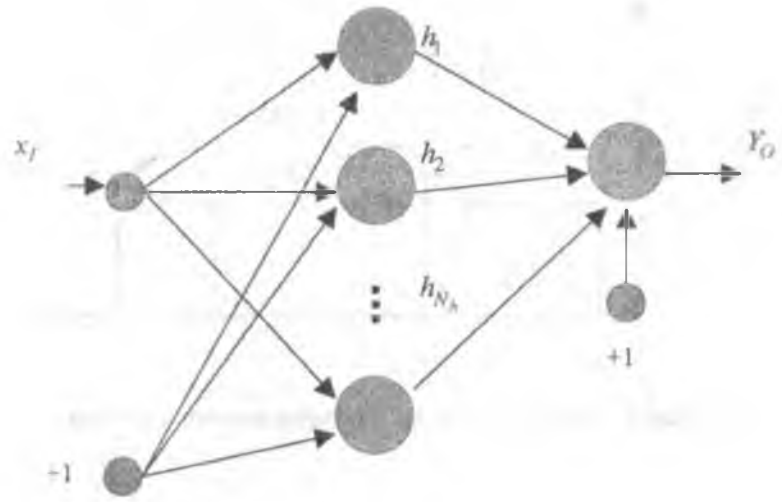


Figure 4. The MLP architecture

6. The production phase for RNN, in which the sensor errors are defined for any moment of time $t > t_n$.

4.1: The MLP Architecture and Training for Approximation

As network architecture for approximation was accepted third-layered multilevel neural network containing one hidden layer of nonlinear units and a single output linear unit, as shown on Fig. 4. The output activity of the neural network is defined by expression:

$$Y = \sum_{i=1}^{N_h} w_{i0} h_i - s_0 \tag{1}$$

where N_h - number of units of the hidden level, h_i - output activity of hidden units, s_0 - threshold for output unit, w_{i0} - weights from hidden input units i to the output unit. The output activity of the hidden units is defined as:

$$h_j = g(w_{ij} x_I + s_j) \tag{2}$$

where x_I is the input element, w_{ij} - weights from input unit to hidden units j and s_j - thresholds of the hidden units. In hidden units is used sigmoid transfer function, defined as $g(x) = \left((1+e^{-x}\right)^{-1}$ .The most popular training algorithm for multilevel perceptrons is backpropagation. This algorithm is based on gradient descent method and consists of fulfilment of an iterative procedure of updating weights and thresholds for each training exemplar p of training set under following rule:

$$\Delta w_{ij}(t) = -\alpha \frac{\partial E^p(t)}{\partial w_{ij}(t)} \tag{3}$$

$$\Delta s_j(t) = -\alpha \frac{\partial E^p(t)}{\partial s_j(t)} \tag{4}$$

where $\frac{\partial E^p(t)}{\partial w_{ij}(t)}$, $\frac{\partial E^p(t)}{\partial s_j(t)}$ - gradients of error function on training iteration t for training exemplar p, $p \in \{1, P\}$, P is the size of the training set;

$$E^p(t) = \frac{1}{2}\left(Y_O^p(t) - D_O^p\right)^2 \tag{5}$$

$Y_O^p(t)$ - network output activity on training iteration t for training exemplar p, D_O^p - desirable value of a network output for training exemplar p. During training there is the reduction process of the total network error:

$$E(t) = \sum_{p=1}^{P} E^p(t) \tag{6}$$

For improvement of network training parameters and removal defects of classical back propagation algorithm, connected with empirical selection of a constant training step, use the steepest descent method for calculation of an adaptive training step, according to it:

$$\begin{cases} \Delta w_{ij}(t) = -\alpha^p(t) \dfrac{\partial E^p(t)}{\partial w_{ij}(t)}, \\ \Delta s_j(t) = -\alpha^p(t) \dfrac{\partial E^p(t)}{\partial s_j(t)}, \\ \alpha^p(t) = \min\left\{E^p\left(w_{ij}(t+1), s_j(t+1)\right)\right\} \end{cases} \tag{7}$$

where $\alpha^p(t)$ - step value, adapted on each training iteration t for each external vector p.

According to expression (7) the formulas for calculation of adaptive step for sigmoid and linear functions of activation were obtained.

For linear transfer function the adaptive training step is defined by expression:

$$\alpha^p(t) = \frac{1}{\sum_{j=1}^{N_h} (h_i^p(t))^2 + 1} \quad (8)$$

where $h_i^p(t)$ - elements of input activity of the linear unit at the time t for the training vector p.

For sigmoid transfer function adaptive training step is computed as:

$$\alpha^p(t) = \frac{4}{1+(x_i^p)^2} \times \frac{\sum_{j=1}^{N_h} (w_{j0})^2 h_j^p(t)(1-h_j^p(t))}{\left(\sum_{j=1}^{N_h} (w_{j0})^2 (h_j^p(t))^2 (1-h_j^p(t))^2\right)} \quad (9)$$

where $w_{i0}(t)$ - weights from hidden units to output unit, adapted on training iteration t.

At training the network is actual problem of initialization of neural network units weights. The speed, accuracy and stability of training depends on it in many respects, especially if is used training sets with analogue values. Usually initialization consists of appropriation to weights and thresholds of units of the random evenly distributed values from some range: $w_{ij} = R(c,d), s_j = R(c,d)$. Upper and low bounds of this range is defined empirically. In our paper the following procedure of delimitation of a range for sigmoid function is offered:

$$w_{Ic} = \frac{\bar{x}_I}{(x_I)^2 + 1} \ln\left(\frac{c}{1-c}\right) \quad (10)$$

$$w_{Id} = \frac{\bar{x}_I}{(x_I)^2 + 1} \ln\left(\frac{d}{1-d}\right) \quad (11)$$

where $\bar{x}_I$ is expectation of the external activity on the MLP input, $\bar{D}_0$ is expectation of the external activity of desired output of the network, w_{Ic}, w_{Id} are bounds of the weights range, c, d are upper desired value and low desired value of the output activity for nonlinear units.

For linear output unit we proposed to calculate parameters w_{ic}, w_{id} as:

$$w_{ic} = \frac{c}{N_h c^2 + 1} \bar{D}_0 \quad (12)$$

$$w_{id} = \frac{d}{N_h d^2 + 1} \bar{D}_0 \quad (13)$$

The weights and thresholds of the units are initialized as:

$$w_{ij} = R(w_{ic}, w_{id})$$
$$s_j = R(w_{ic}, w_{id}) \quad (14)$$

For stabilization of the training procedure is used the following algorithm of *level-by-level* training:

1) Calculate upper and lower bounds of the weights range, using expressions (10), (11) for hidden units and expressions (12), (13) for output unit. Update weights and thresholds according to expressions (14).

2) For the training vector p calculate output activity $Y_O^p(t)$ of the neural network.

3) Calculate an error of an output unit.

4) Update weights and thresholds *only* for the output unit according to expressions (7), using adaptive step (8).

5) Calculate the error of units of the hidden level for the network with the *updated weights for an output level.*

6) Update weights and thresholds of units of the hidden level, using adaptive step (9) for sigmoid transfer function.

The application of this algorithm has allowed to stabilize learning process of the recurrent neural network with varied functions of activation and considerably to reduce time of training.

For simulation is used the time series of sensor errors, described by function $f(\tau) = |\tau + \sin(3\tau)$. Computing experiments for sensor error approximation by MLP shown in the Table 4.1.

Training set size	80
Number of hidden units	6
Parameters c, d (for hidden units)	0.1, 0.9
Total mean square training error	$1.92*10^{-5}$
Number of approximation steps	195
Approximation error in percentage	1.89%

Table 1. Sensor error approximation results by MLP

One can see on the table 1, that MLP approximate the function with small error.

Let's examine the CPNN for approximation.

4.2: The CPNN Architecture and Training for Approximation

The architecture used of CPNN is represented in the Fig 5. It consists of three layers. The hidden layer consists of Cohonen neurons (not shaded circles) and nonlinear neurons (shaded circles), which amount is equaled among themselves. Such neurons will derivate pairs, and each Cohonen neuron in a pair has horizontal connection with a nonlinear neuron, appropriate to it, as shown Fig. 4. Besides all neurons of a hidden layer are connected to output neuron, which has linear function of activation. Let's use sequential numbering of pairs of neural elements. Let's designate through w_{Ij} and u_{Ij} accordingly weight factors of Cohonen neurons and nonlinear neurons of the hidden layer. Then the output value h_j of j'th nonlinear hidden neuron can be defined as

$$h_j = F(y_j u_{Ij} x_I) \quad (15)$$

where y_j - output of a Cohonen neuron in j-th pair of neurons, F - hyperbolic tangent. The output value of j-th Cohonen neuron is equaled to one, if this neuron is a winner, and zero otherwise. For definition of a neuron - winner the Euclidean distance is used:

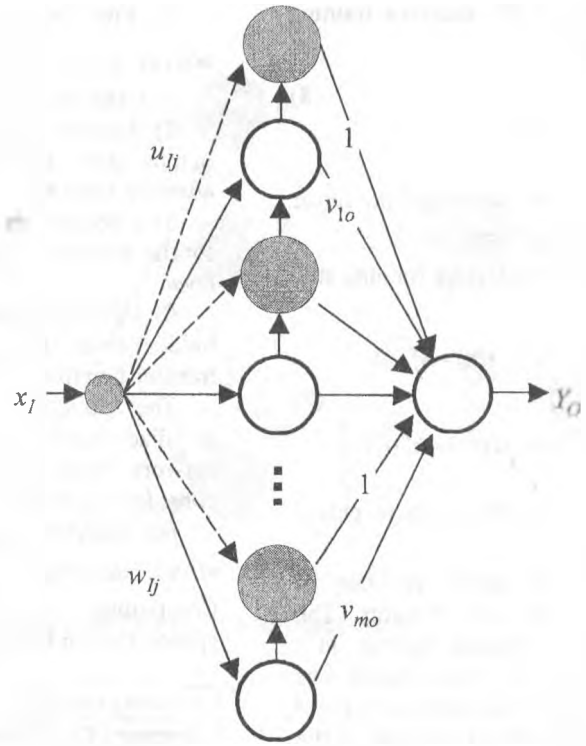


Figure 5. The counterpropagation neural network architecture

$$D_j = |X^p - W_j| \tag{16}$$

In the correspondence with it:

$$y_k = \begin{cases} 1, if\ D_k = \min|X^p - W_j| \\ 0, otherwise \end{cases} \tag{17}$$

The weights from nonlinear neurons of the hidden layer to output neuron are equaled to one.

If a neuron - winner in the hidden layer has number k, the output of the neural network is equal:

$$Y_O = v_k + h_k \tag{18}$$

The amount m of neurons of the hidden layer is selected equal:

$$m = P - 1 \tag{19}$$

where P -size of the training set.

The training of CPNN is made on the following algorithm.

1. The set-up a Cohonen weights by the empirical rule is made:

$$w_{lj} = x_i + \frac{x_{i+1} - x_i}{3} \tag{20}$$

where w_{lj} - weight of the i -th of the Cohonen neuron, x_i - is i -th component of the training set ordered on increase, $i = \{1,...,m\}$.

2. Further in a cycle the set-up a weights of remaining neural elements is executed. On an input of the neural network values of the training set sequentially move and the following operations are made for each value:

- The value of an output of the neural network Y_O^p and number k of a neuron - winner of a Cohonen layer is calculated.
- The set-up of an appropriate neuron of the output layer is made:

$$\Delta v_k(t) = -\alpha(Y_O^p - D_O^p) \tag{21}$$

where D_O^p - target value of CPNN normalized in a range [-1, 1], $\alpha = 0.01$ - training step.

- The appropriate neuron of the hidden layer is set up:

$$\Delta u_{lk}(t) = -\alpha(Y_O^p - D_O^p) \cdot (1 - (h_k^p)^2) x_l^p \tag{22}$$

3. The training total root-mean-square error is calculated:

$$E = \frac{1}{2} \sum_{p=1}^{P} (Y_O^p - D_O^p)^2 \tag{23}$$

The steps 2 and 3 are repeated before stabilization of training error.

Computing experiments for sensor error approximation for various sizes if training set by CPNN shown in the table 2.

Training set size	80
Total mean square training error	$1.587*10^{-5}$
Number of approximation steps	195
Approximation error in percentage	2.58%

Table 2. Sensor error approximation results by CPNN

4.3: The RNN Architecture and Training for Time Series Prediction

As basic network architecture for prediction of the sensor error was accepted fully connected third-layered recurrent neural network containing one hidden layer of nonlinear units and a single output linear unit, as shown in the Fig. 6. The output activity of the neural network is defined by expression:

$$Y^r = \sum_{i=1}^{N_h} w_{i0} h_i^r - s_0 \qquad (24)$$

where N_h - number of units of the hidden level, h_i^r - output activity of hidden units at the moment r, s_0 - threshold for output unit, w_{i0} - weights from hidden input units i to the output unit.

The output activity of the hidden units on the current moment r for training exemplar p is defined as:

$$h_j^r = g\left(\sum_{i=1}^{N_I} w_{ij} x_i^r + \sum_{k=1}^{N_h} w_{kj} h_k^{r-1} + w_{0j} Y^{r-1} + s_j\right) \qquad (25)$$

where x_i^r is a i'th element of the input vector x^r, P is size of the training set, N_I - size of an input vector, w_{ij} - weights from external units i to hidden units j, w_{kj} - weights from hidden units k to hidden units j, $h_k^{r-1}(t)$ - output activity of a hidden unit k for the previous moment of time $r-1$, w_{0j} - weight to the hidden units from an output unit, $Y^{r-1}(t)$ - network output activity for the previous moment of time $r-1$ and s_j - thresholds of the hidden units.

In this work we use two types for hidden units transfer function. At one case is used non-standard logarithmic function $g(x) = \ln\left((x + \sqrt{x^2 + a})/\sqrt{a}\right)$. $(a > 0)$. The choice by this transfer function is stipulated by that it is unlimited on all define area. It allows better to simulate and predict complex non-stationary processes. The parameter a defines declination of the activation function (see Fig 7). At other case in hidden units is used sigmoid transfer function, defined as $g(x) = \left((1 + e^{-x}\right)^{-1}$

For logarithmic activation function the estimate of an adaptive training step can be received by the following expression:

$$\alpha^p(t) = \frac{1}{\left(1 + \sum_{i=1}^{N_I} (x_i^p)^2 + \sum_{k=1}^{N_h} (h_k^{p-1}(t))^2 + (Y^{p-1}(t))^2\right)} \times \frac{\sqrt{a}\sum_{j=1}^{N_h} (w_{j0})^2 \left(\sqrt{(B_j^p(t))^2 + a}\right)^{-1}}{\left(\sum_{j=1}^{N_h} (w_{j0})^2 \left((B_j^p(t))^2 + a\right)^{-1}\right)} \qquad (26)$$

where

$$B_j^p(t) = \sum_{i=1}^{N_I} w_{ij}(t) x_i^p + \sum_{k=1}^{N_h} w_{kj}(t) h_k^{p-1}(t) + w_{0j}(t) Y^{p-1}(t) + s_j$$

is weighed sum of inputs of the hidden unit j.

For standard sigmoid transfer function adaptive training step is computed as:

$$\alpha^p(t) = \frac{4}{\left(1 + \sum_{i=1}^{N_I} (x_i^p)^2 + \sum_{k=1}^{N_h} (h_k^{p-1}(t))^2 + (Y^{p-1}(t))^2\right)} \times \frac{\sum_{j=1}^{N_h} (w_{j0})^2 h_j^p(t)(1 - h_j^p(t))}{\left(\sum_{j=1}^{N_h} (w_{j0})^2 (h_j^p(t))^2 (1 - h_j^p(t))^2\right)} \qquad (27)$$

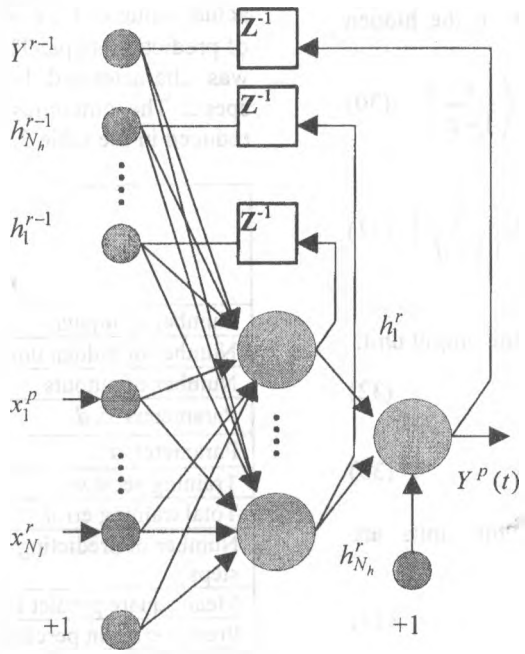


Figure 6. The recurrent neural network architecture

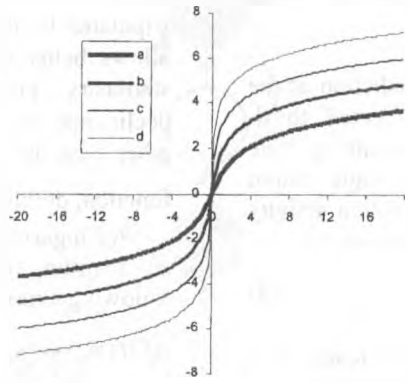


Figure 7. The logarithmic activation function for various parameters a:
a) $a = 1.0$, **b)** $a = 0.1$, **c)** $a = 0.01$, **d)** $a = 0.001$

For adaptive initialization of weights and thresholds in RNN before training is used the next expressions:

1. For logarithmic activation function in the hidden units:

$$w_{ic} = \frac{\bar{x}_i}{\sum_{k=1}^{N_i}(\bar{x}_k)^2 + N_h c^2 + \bar{D}_0^2 + 1} \cdot \frac{\sqrt{a}e^c - \sqrt{a}e^{-c}}{2} \quad (28)$$

$$w_{id} = \frac{\bar{x}_i}{\sum_{k=1}^{N_i}(\bar{x}_k)^2 + N_h d^2 + \bar{D}_0^2 + 1} \cdot \frac{\sqrt{a}e^d - \sqrt{a}e^{-d}}{2} \quad (29)$$

where $\bar{x}_k$ is expectation of the external activity on the k'th inputs, $\bar{D}_0$ is expectation of the external activity of desired output of the network, w_{ic}, w_{id} are bounds of the weights range, c, d are upper desired value and low desired value of the output activity for nonlinear units.

2. For sigmoid activation function in the hidden units:

$$w_{ic} = \frac{\bar{x}_i}{\sum_{i}(\bar{x}_i)^2 + N_h c^2 + \bar{D}_0^2 + 1} \ln\left(\frac{c}{1-c}\right) \quad (30)$$

$$w_{id} = \frac{\bar{x}_i}{\sum_{k=1}^{N_i}(\bar{x}_k)^2 + N_h c^2 + \bar{D}_0^2 + 1} \ln\left(\frac{d}{1-d}\right) \quad (31)$$

3. For linear activation function in the output unit:

$$w_{ic} = \frac{c}{N_h c^2 + 1} \bar{D}_0 \quad (32)$$

$$w_{id} = \frac{d}{N_h d^2 + 1} \bar{D}_0 \quad (33)$$

The weights and thresholds of the units are initialized as:

$$w_{ij} = R(w_{ic}, w_{id})$$
$$s_j = R(w_{ic}, w_{id}) \quad (34)$$

For training of the RNN is used described above level-by-level training algorithm. For simulation were used the time series of sensor errors, taken from MPL (CPNN) approximation results. It size is 195 units. For training were used 105 units of this number. The training was carried out the method of the sliding window. For simulation two types of neural networks were used. One of them contained the sigmoid activation function of the hidden units. In other network the logarithmic function of activation of the hidden units with the parameter $a = 0.01$ was used. Both networks consist of 10 input units, 5 hidden units, 1 output unit. The prediction was carried out on 90 steps forwards. For an estimation of the prediction results is used the mean square predict error computed as:

$$E_{pr}(L) = \frac{1}{L}\sum_{l=1}^{L}\left(\hat{Y}(l) - x(l)\right)^2 \quad (35)$$

where $\hat{Y}(l)$ - predict value for the step l, $x(l)$ - actual value of time series in the moment l, L - total of prediction steps. The training both neural networks was characterized by high accuracy, stability and speed. The outcomes of training and prediction are reduced in the table 3.

	Sigmoid network architect.	Logarithmic network architect.
Number of inputs	10	10
Number of hidden units	5	5
Number of outputs	1	1
Parameters c, d	0.9, 0.1	4.2, -4.2
Parameter a	-	0.01
Training set size	105	105
Total training error	3E-5	4E-5
Number of predicting steps	90	90
Mean square predict error	3.57E-5	1.19E-5
Predict error in percentage	4.34%	3.21%

Table 3. Sensor error prediction experiments by RNNs

5: Conclusion

As it is visible, some operating performances of measuring engineering hinder direct use of neural networks for increase of an exactitude of measuring channels. However use of features of neural networks and organization of their correct interaction with the measuring system allow to overcome originating difficulties and to supply increase of an exactitude of a measurement for want of minor working costs. In this work the unique technology of forecasting of errors of gauges with use of a complicated neural system is circumscribed. The effective methods of neural networks training of different architectures are indicated. The computing experiments with hypothetical datas of errors of sensors demonstrate potential possibilities of application of this intellectual neural system in actual problems

Acknowledgement

Authors would like to thank INTAS, grant reference number INTAS-OPEN-97-0606.

6: References

[1] http://www.honeywell.com/sensing/prodinfo/temperature/catalog/c15_93.pdf

[2] Samsonov G.V., Kits A.I., Kiuzdeni O.A., "Sensors for temperature measurement in industry", *Naukova dumka*, Kiev, 1972

[3] www.fluke.com/products/data_acquisition/hydra/home.asp?SID=7&AGID=0&PID=5308

[4] Iyengar S.S., "Distributed Sensor Network - Introduction to the Special Section", *Transaction on Systems, Man and Cybernetics*, vol. 21, No. 5, 1991, pp 1027-1035

[5] Brignell J., "Digital compensation of sensors", *Scientific Instruments*, vol. 20, No 9, 1987, pp.1097-1102

[6] Sydenham P.H., "Sensing Science and Engineering Group", *Measurement*, vol 14, No 1, 1994, pp.81-87

[7] Lee K., "The Emerging IEEE-P1451 Standards for Smart Sensors and Actuators", *Instrumentation and Measurement Society Newsletter*, 1997, Issue No 136

[8] P. Daponte, D. Grimaldi, "Artificial Neural Networks in Measurements", *Measurement*, vol. 23, 1998, pp.93-115.

[9] C.Alippi, A.Ferrero, V.Piuri, "Artificial Intelligence for Instruments & Applications", *IEEE I&M Magazine*, June 1998, pp.9-17.

[10] Rogelberg I.L., Nushnov A.G., Pokrovskaya G.N. et al., "Stability of the thermo-E.M.F. force of K-type thermocouples at heating in the air at 1200°Ñ temperature", *Hypromettsvetobrabotka*, vol. 24, 1967, pp.54-65

[11] Sachenko A., Milchenko V., Kochan V., Chyrka M., Karachka A., "Experimental researches of the calibrated characteristics of non-stability of K-type thermocouples", *Izmeritelnaja technika*, No 10, 1985, pp.28-29.

[12] Jain A.K., Mao J., Mohiuddin K.M., "Artificial neural networks: a tutorial", *Computer*, March 1996, pp.31-44.

[13] Kroese B. An introduction to Neural Networks.– Amsterdam: Univercity of Amsterdam – 1996.–120p.

Data Fusion Algorithm in Intelligent Distributed Sensor Networks

A. Sachenko, V. Kochan, V. Turchenko, V. Koval

Institute of Computer Information Technologies
Ternopil Academy of National Economy
3 Peremoga Square, 282004 Ternopil UKRAINE
Phone: +380 (352) 33-0830 Fax: +380 (352) 33-0024
E-Mail: sachenko@cit.tane.ternopil.ua

Abstract

The offered data fusion algorithm is based on the averaging information received from reasonable sensors, whose readings always contain errors. The decision can be applied for receiving the most correct result about the object condition that is determined by set of various physical measures (coordinate, height, speed, temperature, etc.) in a space, defensive industry and measuring engineering.

1. Introduction

At the modern stage of engineering development, the surrounded world can be presented as hierarchical totality of automated systems, which elements are linked in networks. The latter contains, as a rule, heterogeneous sensors, which are distributed by means of various criteria: logically, spatially, geographically [1]. Therefore rather important and actual problem of sensor's processing appears in space, defensive industry, electric power industry, measuring engineering, medical area, robotics, chemical industry and in other spheres. Intelligent distributed sensor systems and networks are widely used at the present moment [2, 3]. In most cases the role of intelligent systems functions can be reduced to the fact that they are capable of doing sensor perception of environment and possessing sufficient knowledge (intelligence) to give an adequate reaction to the researched situation or environment. That is why the task of creation data fusion algorithms for reasonable sensors (RS) in intelligent distributed sensor networks is very actual.

As the existing sensor fusion algorithms [4, 5, 6] have some negative sides, the comparative estimation of these algorithms is conducted below. Also the algorithm of mean square weight factors (MSWF) is offered.

2. Sensor fusion algorithms

Let us consider a network of reasonable sensors. Reasonable sensors are sensors, which fulfill some primary processing of an input signal, and the result of processing is represented in digital form. The simple distributed network (Fig. 1) contains set of RS, which are linked among them by the full graph scheme.

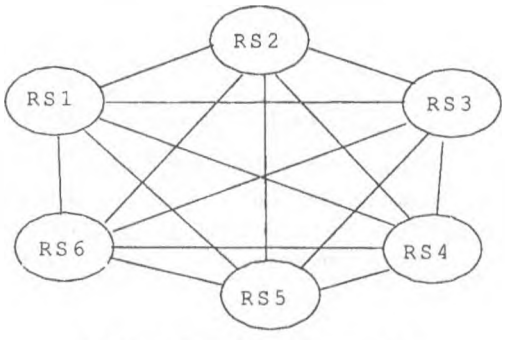


Fig.1. Distributed network of RS

With the reference to radar engineering [1], receiving and processing of information from RS is a process of getting the possible data about object (coordinate, height, speed, course corner, location time etc.), which is in a zone of RS visibility. As RS areas action are crossed the information about one object can arrive from several RS. The data about object that received from RS should be imposed in ideal case. Nevertheless, the coincidences are not

observed in practice because of systematic and casual errors. By virtue of these reasons there are difficulties in data fusion. When sensor data are fused the accuracy and reliability of the received result is decisive criteria of data fusion algorithms functioning.

D. Dolev has presented Byzantine agreement algorithm for the solving of the Byzantine generals problem, which was researched by L. Lamport and his colleagues [7, 5, 8]. The main idea of the Byzantine generals problem solving is the following: if there are N of independent RS, then is supposed such number t of faulty elements (sensors) at which the inequality should be carried out

$$N \geq 3 * t + 1. \qquad (1)$$

That is each RS should be linked with not less than $2 * t + 1$ other RS.

The main principle of data fusion algorithm offered by D. Dolev is firstly checking of Byzantine agreement condition (1). The highest (x_{max}) and lowest (x_{min}) RS values are rejected, then calculated an average value of remained elements (reference value of element)

$$Z_j = \frac{\sum_{i=1}^{N} x_{ij} - (x_{\max j} + x_{\min j})}{N-2},$$

where Z_j - the reference value of element, j - number of RS that sends the data, i - number of RS that receives the data, N – RS numbers, x_{ij} – value which receives i-RS from j-RS.

The resulting value is

$$Z_{rez} = \frac{\sum_{j=1}^{N-1} Z_j}{N}$$

As it is visible from Fig. 2 the algorithm has essential lack. Initial signal (resulting value) value is less correct owing to rejection of the highest and lowest element, than average value of sample (on Fig.2 the average value of sample comes nearer to true value). Therefore such algorithm is correct only in that case, when the value of the highest and (or) lowest element is by rough errors, which deviate the average value from real signal.

In sensor fusion algorithm by S.Mahaney and F.Schneider [4, 6], as against from the previous algorithm, the concepts of accuracy and precision of RS are introduced. If to designate an error through $U_e(t)$ and true signal through $U_t(t)$ then RS readings can be in boundaries

$$U_e(t) - U_t(t) \leq U_t(t) \leq U_e(t) + U_t(t).$$

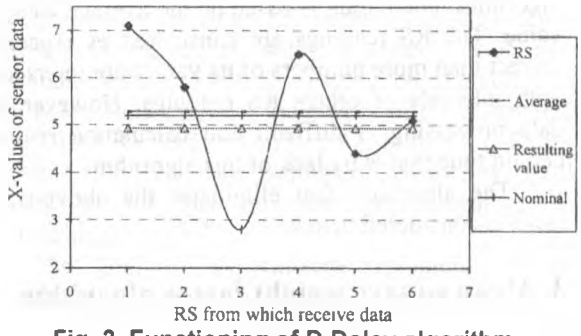


Fig. 2. Functioning of D.Dolev algorithm on example of 6 RS

Therefore, it is possible to set the readings of each RS by an interval of allowable values. S. Mahaney's and F. Schneider's algorithm uses groups of the allowable data. The value is admitted if it in common boundaries for all RS area (this area is formed on the basis of the maximal value among the lowest boundaries of RS accuracy and minimal value among the highest boundaries). If the interval of the RS readings is in common boundaries for all RS of values area, it is considered as admitted, differently its readings are rejected (Fig. 3). It is obvious, that any value that is not admitted can not be correct. The algorithm result is average arithmetic value of the middle of allowable value intervals of the RS readings. This algorithm as against previous will not carry out elimination in the RS readings at the large rejections from average value.

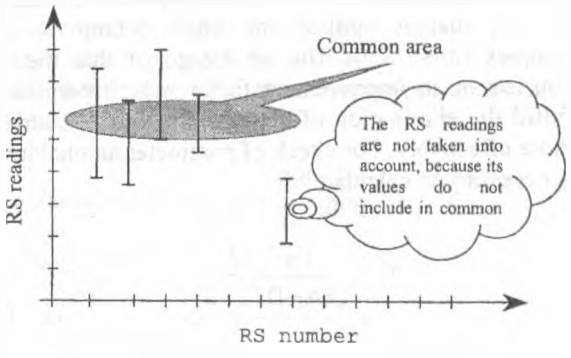


Fig. 3. Common area of sensor readings accuracy

In the Brooks-Iyengar hybrid algorithm [4], there

are groups with N real values and accuracy is determined by distance from unique correct value. As all RS have limited accuracy then its values consist of the highest and lowest boundaries. This algorithm uses the intervals of allowable values of RS readings and takes into account Byzantine agreement problem. The algorithm functioning is based on the average weighed value. The RS readings are considered as especially correct than more numbers of its values are intersected with intervals of others RS readings. However, the data processing is difficult and calculation requires certain time that is by lack of this algorithm.

The algorithm that eliminates the above-stated lacks is considered below.

3. Mean square weight factor algorithm

3.1. Background

The offered algorithm of mean square weight factor is based on assumption that all sensor's values of RS output signals in intelligent distributed sensor network represent by data sample. Conditions described in Byzantine generals problem by D.Dolev [5] is checked firstly. After this the preliminary data processing is executed that applying of statistical methods will be more correct.

The preliminary processing basically consists of elimination of rough errors. It is possible to explain essence of such errors (abnormal or strongly detailed values) by example [9]. Let us to allow that 10% of measurement results represented by abnormal values, differ from average more than $3l$ (l – line segment on axis $\dot{I}\tilde{O}$). If the rest of readings are placed within the l limits then these 10% redouble this estimation at least.

Among set of methods of elimination rougherrors [9], it is necessary to note method of the maximal relative deviation with improvement factor [9, 10] that is applied for small behind volume samples (n <= 25). The advantage of this method consists in an improvement factor, which permits to fulfill the elimination of abnormal values of samples more effectively. For check of parameter anomalous it is necessary to calculate

$$\tau_i' = \frac{|x_{ij} - \bar{x}_i|}{\sqrt{(n-1)/n} * \bar{S}_i} \qquad (2)$$

where x_{ij} - value receiving by ^{3}-RS from j-RS that is checked on anomalous, x_i - average value ^{3}-sample, n-number of sample elements, τ' - quantile of statistics distribution.

$$\bar{S}_i = \sqrt{\frac{1}{n-1}\sum_{j=1}^{n}(x_{ij} - \bar{x}_i)^2}$$

- mean square deviation of the unbiased estimation variance.

Calculated according to (2) values τ'_i is compared to tabulared value τ_{1-p} [9] that is calculated with confidence probability $q=1-p$. If the calculated value $\tau'_i \le \tau_{1-p}$ then it is possible to approve that value not abnormal with probability P. In other case value reject from sample and check anomalous of value carry out repeatedly on rest elements. For large on the volume samples n>25 it is expedient to use the tables of Student distribution [9, 10].

As it is visible from Fig. 4 the abnormal error value significantly hold away the average sample value from true value therefore roughvalue needs to rejected from sample. It is necessary to note that each result of separate measurement is equal to the sum of true value and casual error. Therefore law of measurement result distribution will coincide with the law of casual error distribution.

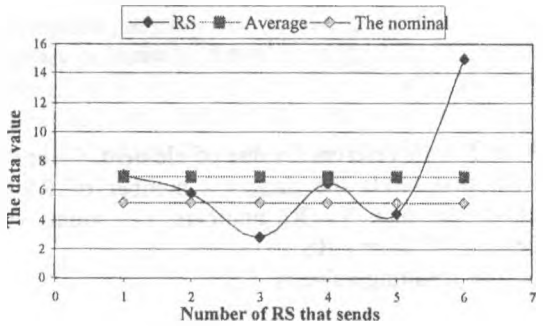


Fig. 4. The schedule of rough errors influence on the result

The accuracy of measurements is estimated by average square law deviation from average arithmetic sample. After elimination of rougherrors the average square deviation of measurement result from average [9] is

$$\sigma = \sqrt{\frac{\sum_{j=1}^{n}(x_{ij} - \bar{x}_i)^2}{n-1}} \qquad (3)$$

From expression (3) it is possible to find estimation of average quadratic error of measurement result [11].

$$\bar{\sigma}_i = \frac{\sigma_i}{\sqrt{n}} \qquad (4)$$

Having average square estimation (4), it is possible to note that the true value is in limits [11, 12]

$$\bar{x}_i - \bar{\sigma}_i \le value \le \bar{x}_i + \bar{\sigma}_i .$$

It is obvious that the greatest value of average square law deviation, then measuring device bring large error into measurement result. Therefore, as against the previous methods it is necessary to use not arithmetic mean, but average weighed value (taking into account factors of each value importance) for averaging of sensors data

$$\bar{X}_i = \frac{\sum_{j=1}^{n} k_{ij} x_{ij}}{\sum_{j=1}^{n} k_{ij}}$$

where k_{ij} - factor of importance of j – parameter that is accepted by i-RS. As last uses back proportional to square of measurement error

$$k_{ij} = \frac{1}{(x_{ij} - \bar{x}_i)^2}$$

When all average weighed values ^{3}-RS are found one uniform resulting value is calculated as arithmetic mean of weighed average

$$\bar{X}_{rez} = \frac{\sum_{i=1}^{n-t} \bar{X}_i}{n-t} \qquad (5)$$

where n – number of RS, t - quantity of fault elements.

Thus, from averaging result founded average estimation has smaller casual error than separate values (by that it is founded).

3.2. Experimental results

For estimation of data fusion algorithm quality we shall fulfill experiments with the help of imitating model. In experiments is used 6 RS which receive data about measurement object. Thus there is one fault RS, which sends to another RS various readings about measurement signal and with the greatest error. The true signal value is 5.230. For estimation of algorithms quality we shall consider their work at measurements errors: first RS ±2.5, second ±0.41, third ±1.7, fourth ±2.9 and fifth ±1.8. The readings of sixth RS we shall consider as fault, therefore measurement error of this RS true signal is ±5.4.

The goal of experimental researches is to consider functioning of data fusion algorithms with use of measurement errors distributed on the various distributivelaws. Thus for elimination of the received results chance and partial cases it is necessary to fulfill not less than 10 experiments of imitating model.

Let us consider algorithms functioning at the equal distributive law of measurement errors. The table with true signal values (Table 1.1), table with deviation from true signal values (Table 1.2) and the table with relative deviations of all algorithm values from MSWF algorithm (Table 1.2) are showed below. From these tables it is necessary to note that the MSWF algorithm has shown the best results (greatest quantity of the smallest deviations from true signal; there is no one worse result; best results of relative deviations) in comparison with other algorithms (Fig. 5). Concerning cases, when the results of MSWF algorithm worse from the results of other algorithms, it is necessary to node that their size is insignificant in comparison with the best results. The quantity of the worse results is insignificant in comparison with the best results also (one result from ten for algorithm "D.Dolev" and three results from ten for "R.Brooks and S.Iyengar" and "S.Mahaney" algorithm).

The results of functioning of all algorithms at the normal distributive law of measurement errors are presented in the Tables 2.1 - 2.3.

Table 1.1.
The calculated values of true signal

	1	2	3	4	5	6	7	8	9	10
True signal	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300
Dolev	5,1592	5,1485	5,3249	5,5192	5,6599	5,3535	4,9532	5,9629	5,5192	5,1485
Mahaney	5,1110	5,1736	5,4982	5,5395	5,7466	5,2241	5,0750	5,7461	5,5396	5,1736
Brooks	5,3309	5,0693	5,3748	5,1509	5,4673	5,0979	5,4787	5,3187	5,1509	5,0693
MSWF	5,1704	5,1855	5,3918	5,2714	5,5225	5,2619	5,0690	5,8880	5,2714	5,1855

Table 1.2.
Deviation from true signal values

	1	2	3	4	5	6	7	8	9	10
Dolev	0,0708	0,0815	0,0950	0,2892	0,4299	0,1235	0,2768	0,7329	0,2892	0,0815
Mahaney	0,1190	0,0564	0,2682	0,3095	0,5166	0.0059	0,1550	0,5161	0,3096	0,0564
Brooks	0,1010	0,1607	0,1448	0,0791	0,2373	0,1321	0,2487	0,0887	0,0791	0,1607
MSWF	0,0596	0,0445	0,1618	0,0414	0,2925	0,0319	0,1610	0,6580	0,0414	0,0445

Table 1.3.
Relative deviations of all algorithms values from MSWF algorithm

	1	2	3	4	5	6	7	8	9	10
Dolev	1,1879	1,8315	0,5869	6,9855	1,4697	3,8715	1,7193	1,1138	6,9855	1,8315
Mahaney	1,9966	1,2674	1,6576	7,4758	1,7662	0,1850	0,9627	0,7843	7,4783	1,2674
Brooks	1,6941	3,6112	0,8949	1,9106	0,8113	4,1411	1,5447	0,1348	1,9106	3,6112

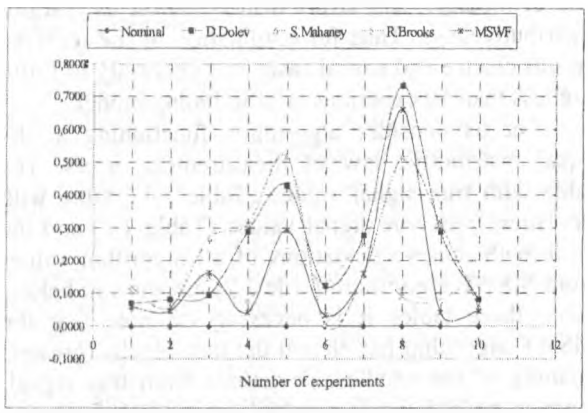


Fig. 5 Diagram of deviation of true signal from its real value (equal distribution law)

The analysis of these tables has shown that the MSWF algorithm is more effective. The number of the best results (minimal deviations from true signal) is greatest (Fig. 6). Also the MSWF algorithm is better taking into account the worst cases, when the deviations of the resulting readings from true signal is greatest (numbers of the worse results at MSWF algorithm is equal 2 and in "D.Dolev" algorithm is 4).

The experiments were fulfilled with use of the exponential and Simpson distributive laws of measurement errors. At using of the exponential distributive law the most stable value receives "R.Brooks and S.Iyengar" algorithm. Its results have little change in comparison with true signal, though it shows the worst results on occasion. At using of the Simpson distributive law the results of MSWF algorithm are better (greatest quantity of the least deviations from true signal and rather not plenty of the greatest deviations). If consider relative deviations then the results of MSWF algorithm in 4.5 times is best in comparison with other algorithms.

The experimental results of data fusion algorithms functioning have confirmed, that at using of equal, normal, exponential and Simpson distributive laws the most preferable is MSWF algorithm. The results of the given algorithm are best at equal, normal and Simpson distributive laws. At exponentially distributed measurement errors it has worse results where high stability has shown "R.Brooks and S.Iyengar" algorithm. Therefore, the developed MSWF algorithm more effective calculates the correct value at the existence of the fault data. Using of this algorithm can improve accuracy and precision in many distributed applications. The MSWF algorithm will be used in intelligent distributed sensor network [3] where sensor errors are mainly distributed on normal distributive law.

Table 2.1.
The calculated values of true signal

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Real signal	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300	5,2300
Dolev	5,2083	5,2547	5,3558	5,7200	5,3289	4,9702	5,2167	5,3159	5,3327	4,9898	4,9368	5,2815	5,2696	5,4345
Mahaney	5,2419	5,1941	5,2930	5,6596	5,2819	5,0843	5,2923	5,4470	5,2692	4,8184	4,9481	5,3165	5,2670	5,3986
Brooks	5,0828	5,1229	5,1841	5,5279	4,9373	5,2438	5,1169	5,1225	5,3586	4,9534	5,0784	5,3080	5,0144	5,4218
MSWF	5,2497	5,1400	5,2185	5,6824	5,2516	5,0761	5,1559	5,2672	5,2391	4,9727	5,0632	5,3796	5,2763	5,5818

Table 2.2.
Deviation from true signal values

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Dolev	0,0216	0,0247	0,1258	0,4900	0,0989	0,2597	0,0132	0,0859	0,1027	0,2401	0,293194	0,051528	0,039608	0,204585
Mahaney	0,0119	0,0358	0,0630	0,4296	0,0519	0,1456	0,0623	0,2170	0,0392	0,4115	0,281898	0,086592	0,037052	0,168679
Brooks	0,1471	0,1070	0,0458	0,2979	0,2926	0,0138	0,1130	0,1074	0,1286	0,2765	0,15153	0,07801	0,215508	0.191833
MSWF	0,0197	0,0899	0,0114	0,4524	0,0216	0,1538	0,0740	0,0372	0,0091	0,2572	0,166736	0,149691	0,046336	0,351878

Table 2.3.
Relative deviations of all algorithms values from MSWF algorithm

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Dolev	1,0979	0,2750	11,035	1,0831	4,5610	1,6888	0,1786	2,3049	11,190	0,9334	1,758438	0,34423	0,854811	0,581408
Mahaney	0,6068	0,3982	5,5247	0,9496	2,3950	0,9466	0,8421	5,8206	4,2772	1,5998	1,690689	0,578468	0,799645	0,479368
Brooks	7,4522	1,1901	4,0226	0,6586	13,496	0,0901	1,5265	2,8812	14,004	1,0750	0,908801	0,521138	4,651006	0,545169

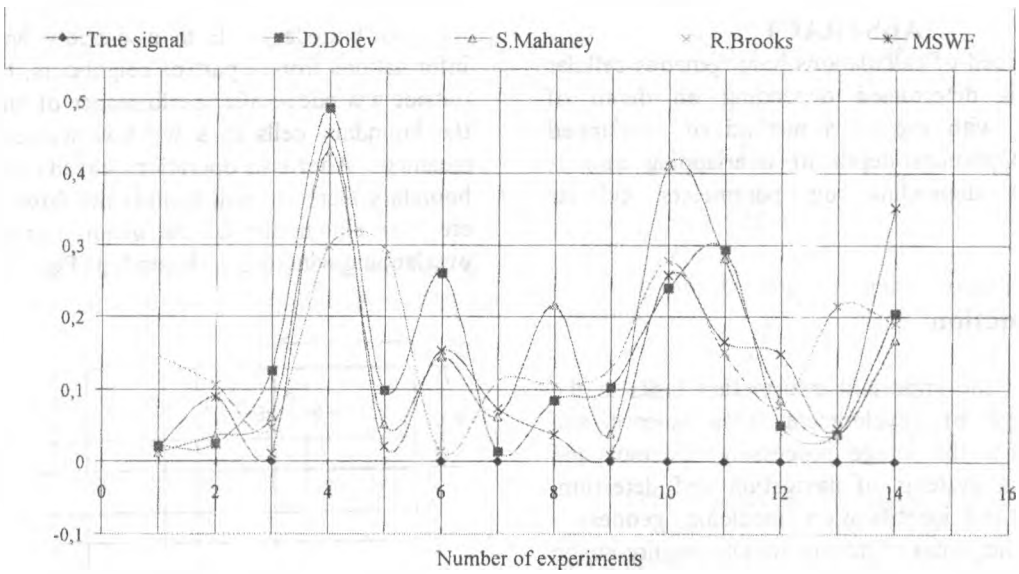


Fig. 5 Diagram of deviation of true signal from its real value (normal distribution law)

4. Conclusion

Algorithm of mean-square weight factor is offered and experimentally checked. It has shown the best results at calculation of values from sensors with different data disorder.

References

[1] Dryshinin V., "Handbook of radar-tracking engin ", Moscow, *Voennoe izdatelstvo*, 1967, 751p.

[2] Iyengar S., Member S., "Distributed Sensor Network – Introduction to the Special Section", *Transaction on Systems, Man and Cybernetics*, vol. 21, No. 5, 1991.

[3] Sachenko A., Kochan V., Turchenko V., "Intelligent Distributed Sensor Network", *Proc. of IMTC/98*, St.Paul, USA, vol.I, 1998, pp.60-66.

[4] Brooks R., Iyengar S., "Robust Distributed Computing and Sensing Algorithm", *Computer*, June 1996, pp.53-60.

[5] Dolev D., "The Byzantine Generals Strike Again", *J.Algorithms*, 1982, pp.14-30.

[6] Mahaney S. and Schneider F., "Reaching Approximate Agreement in the Presence of Faults", *J.ACM*, July 1986, 516p.

[7] Barborak M., Malek M., Dahbura A., "The consensus problem in fault tolerant computing", *ACM Computing Surveys*, Apr. 1986, 349p.

[8] Lamport L., Shostak R., and Pease M, "The Byzantine Generals Problem ', *ACM Trans. Program. Lang. Syst.*, July 1982, pp.382-401.

[9] Lvovsky E., "Statistical methods of the formula construction", Moscow, *Vyshaja shkola*, 1988, 239p.

[10] Kuzin O., "The basic of cybernetics", vol. 2, Moscow, *Energia*, 1979, 586p.

[11] Kasandrova O., Lebedev V., "Processing of supervision results", Moscow, *Nauka*, 1970.

[12] Kozlov T., Ovsienko V., Smirnsky V., "Course of common statistic theory", Moscow, *Statistica*, 1965.

RESEARCH of OPTIMUM DEPTH of OVERLAPPING in CELLULAR AUTOMATA

V.A.Prytkov
220600, Belarus, Minsk, P.Brovki,6, BSUIR, dep. of computers
e-mail: kan@cit.org.by

Keywords: image processing, homogeneous cellular automata, overlapped windows, depth of overlapping.

ABSTRACT

The speed of calculations homogeneous cellular automata is determined depending on depth of overlapping with use of a method of overlapped windows. Optimum depth of overlapping also is investigated depending on parameters cellular automata.

1: Introduction

One of the important information tasks at the present stage of development of a science and engineering is the image processing. Cosmo- and aeroshooting, systems of navigation and detection, recognition and identification, medicine, geodesy - these and other areas of human activity require image processing.

In most cases images are represented as two-dimensional matrix by M x N. For processing such images are widely used cellular logic and cellular automata (CA).

The widespread matrixes of operational elements (OE) CA - four-coherent and eight-coherent.

OE in a matrix are connected among themselves by local communications. OE carries out some set of operations of cellular logic, when the result of operation is defined as logic function of meaning of the element and its neighbours.

One of problems of the image processing on CA is connected that the image in most cases has the sizes exceeding the sizes of a matrix OE. In this case image process by a successive - parallel way, breaking them on separate window [1]. The image by the size M x N is processed on a matrix OE by the size m x n (size of a window), and m < M and n < N.

As boundary OE in a window have not the informations from a part of neighbours, taking place outside a window, after performance of one operation the boundary cells in a window will contain false meanings. After two operations already two layers of boundary elements will contain the false information etc. For elimination of the given phenomenon use overlapping windows with depth p (Fig. 1).

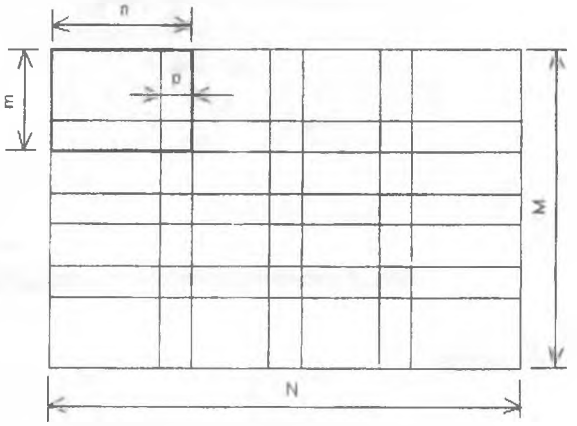


Figure 1.Processing by a method of overlapping windows.

Depending on a kind of processing the various variants of use of overlapped windows are possible [2,3]. Actually the CA can use the various kinds of processing and the size of a source image can be changed in a wide range. Therefore the problem of choice of depth of overlapping depending on parameters of a system is of interest.

In work the efficiency of use of the given method is considered depending on the sizes of the image, matrix OE, depth of overlapping and reloading speed of a matrix OE. Optimum depth of

overlapping also is considered depending on parameters of system.

2: Dependence of time of processing on depth of overlapping

Let's define time required for image processing, if m x n - size of a window, M x N - size of the image, p - depth of overlapping, k - number of operations, which are necessary to execute, t_{op} - the time of performance of one operation (is supposed, that any operation is carried out for identical time), t_{rl} - time spent on reloading CA with transition from one window to another.

Let V_h - number of windows on a horizontal. Then

$$(V_h - 1)\cdot(n-p)+n = N$$
$$V_h = \text{int}^+[(N-p)/(n-p)] \quad (1)$$

Here $\text{Int}^+[x]$ means nearest greater whole. Similarly, the number of windows on a vertical V_v is defined by

$$V_v = \text{int}^+[(M-p)/(m-p)] \quad (2)$$

The complete number of windows V is determined by

$$V = V_h \cdot V_v \quad (3)$$

The number of operations G, which can be executed without occurrence of the erroneous data, found by

$$G = \text{int}[\frac{p}{2}] \quad (4)$$

Here int[x] means the whole part of number. The number of passes H under the image specified by

$$H = \text{int}^+[\frac{k}{G}] \quad (5)$$

The time of processing of the image T is given by

$$T = t_{op}\cdot k\cdot V + t_{rl}\cdot V \quad H \quad (6)$$

By substituting (1-5) in (6), we shall receive

$$T = \text{int}^+[\frac{N-p}{n-p}]\cdot\text{int}^+[\frac{M-p}{m-p}] \quad (t_{op}\cdot k + t_{rl}\cdot\text{int}^+[\frac{k}{\text{int}[\frac{p}{2}]}]) \quad (7)$$

Let's determine expenses of a time T_n if not to make overlapping windows:

$$V_h = \text{int}^+[\frac{N}{n}]$$
$$V_v = \text{int}^+[\frac{M}{m}] \quad (8)$$
$$H = k$$

By substituting (3) and (8) in (6), we shall receive

$$T_n = \text{int}^+[\frac{N}{n}]\cdot\text{int}^+[\frac{M}{m}]\cdot k\cdot(t_{op}+t_{rl}) \quad (9)$$

Let's express parameters M, N, m, p through n, and t_{rl} through t_{op}, by entering corresponding coefficients and assume $n \le m$ and $t_{rl} \ge t_{op}$:

$$\begin{aligned} m &= S_1\cdot n, & S_1 &\ge 1 \\ N &= S_2\ n, & S_2 &\ge 1 \\ M &= S_3\ n, & S_3 &\ge 1 \\ p &= S_4\cdot n, & 0 < S_4 &< 1 \\ t_{rl} &= S_5\cdot t_{op}, & S_5 &\ge 1 \end{aligned} \quad (10)$$

The restrictions on S_2 and S_3 define the size of the image more or equal to a size of matrix OE. The restriction on S_4 defines, that the depth of overlapping lays in limits from 0 up to the short party of a matrix OE. The restriction on S_5 specifies, that the reloading time of a matrix OE is not less time of performance of any other operation. Besides that the variables m, n, M, N, p are integers.

Then the expressions (7) and (9) will be copied as

$$T = \text{int}^+[\frac{S_2 - S_4}{1 - S_4}] \cdot \text{int}^+[\frac{S_3 - S_4}{S_1 - S_4}] \cdot t_{op} \cdot (k + S_5 \cdot \text{int}^+[\frac{k}{\text{int}[\frac{n \cdot S_4}{2}]}])$$

$$T_n = \text{int}^+[S_2] \cdot \text{int}^+[\frac{S_3}{S_1}] \cdot k \cdot t_{op} \cdot (1 + S_5) \qquad (11)$$

What to estimate as far as the processing with overlapped windows is effective, we shall calculate the attitude R = T/Tn:

$$R = \frac{\text{int}^+[\frac{S_2 - S_4}{1 - S_4}] \cdot \text{int}^+[\frac{S_3 - S_4}{S_1 - S_4}] \cdot (k + S_5 \cdot \text{int}^+[\frac{k}{\text{int}[\frac{n \cdot S_4}{2}]}])}{\text{int}^+[S_2] \cdot \text{int}^+[\frac{S_3}{S_1}] \cdot k \cdot (1 + S_5)} \qquad (12)$$

So, we have deduced dependence of efficiency of use of a method of the overlapped windows for homogeneous CA from parameters of system.

For determination of optimum depth of overlapping it is necessary to resolve the equation $R'(S_4)=0$. However even if to drop all non-continuous functions int˙ the problem is reduced to a solution of a equation of fourth grade. On this reason optimum depth of overlapping was simulation.

3: Results of modeling

The estimation of efficiency of depth of overlapping depending on parameters of system was simulated on the computer. For it the meanings S_1, S_2, S_3, S_5, k, n were fixed. The meaning of depth of overlapping p was changed from 1 up to n with a step in 1 pixel. For each of p the meaning S_4 and R under the formulas (10) and (12) accordingly was calculated. Parameters S_1, S_2, S_3, S_5, k, n further were varied and again meanings S_4 and R for each of p was calculated. The results of modeling are submitted on fig.2.

In a figure the family of dependences R (S_4) is submitted with n = 128, S_1 = 1, S_2 = 8 and S_3 = 10. This parameters approximately corresponds to real system (the size matrix of OE 128x128 elements and the size of image 1024x1280 pixels). The diagrams are given for S_5 = 1 and k = 10, for S_5 = 10 and k = 10, for S_5 = 10 and k = 100, for S_5 = 100 and k = 10, for S_5 = 100 and k = 100. Such values of S_5 and k were taken only with goal to demonstrate that in some cases the use of overlapped windows is increasing speed of image processing.

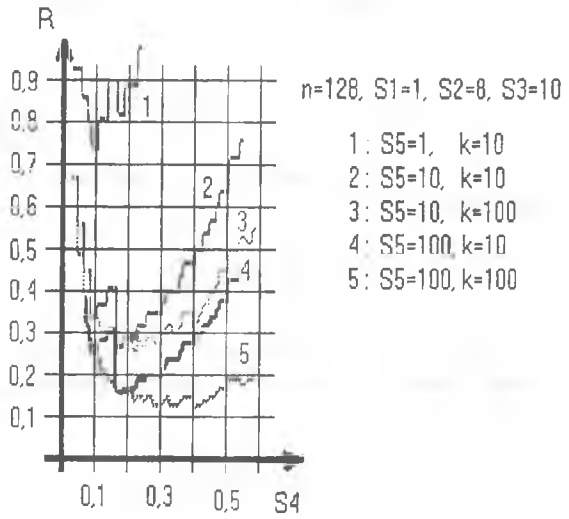


Figure 2. Family of dependences R (S4) with n = 128, S1 = 1, S2 = 8 and S3 = 10.

From the diagrams it is visible, that with increase of reloading time CA S_5 the efficiency of use of overlapping raises. So, with S_5 = 1 R is not lowered below 0.7, and with S_5 = 10 can be reduced less than 0.3. Besides that with increase S_5 optimum depth of overlapping (the meaning of depth of overlapping p with which is observed a minimum R) is increasing too.

The change of number of carried out operations k influences the diagram differently. With increase k the optimum depth of overlapping practically does not change, however right side of the diagram becomes essential horizontally. Thereof the size R becomes close to minimal for a wide range of depth of overlapping p.

The numerical meanings of optimum depth of overlapping S_4, size R achieved with the given depth, and range of depth of overlapping, for which $R \leq 1.1R_{min}$ are shown in table 1. R_{min} designates the minimal meaning R for given S_1, S_2, S_3, S_5, k, n with change p from 1 up to n-1.

Optimum depth of overlapping in a number of cases is submitted by range of meanings. It not that other as influence of non-continuos function int^+. For the same reason there are a few intervals of close to minimal meanings S_4 represented in the last column.

Table.1.

Meanings of optimum depth of overlapping S_4, and R achieved with the given depth, and range of depth of overlapping, for which $R \leq 1.1R_{min}$.

with n=128, S_1=1, S_2=8, S_3=10	Optimum depth of overlapping S_4^{opt}	R with $S_4=S_4^{opt}$	Range S_4 with $R \leq 1.1R_{min}$
S_5=1, k=10	0.086-0.094	0.7425	0.070-0.125
S_5=10, k=10	0.164-0.180	0.2727	0.164-0.219
S_5=10, k=100	0.211-0.219	0.2659	0.164-0.219 0.242-0.250 0.273-0.297
S_5=100, k=10	0.164-0.180	0.1634	0.164-0.219
S_5=100, k=100	0.398	0.1287	0.273-0.297 0.320-0.352 0.398-0.414

REFERENCES

1. M.J.B. Duff, T.J. Fountain. Cellular Logic Image Processing. - London, Academic Press, 1986.
2. H.D. Cheng, C. Tong, Y.J. Lu. VLSI Curve Detector. - Pattern Recognition, Vol. 23, N 1/2, 1990.
3. Ming-Hua Chen, Teo Pavlidis. Image Seaming for Segmentation on Parallel Architecture.-.IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 12, N 6, 1990.

On the Detecting of Symmetries of Switching Functions for Efficient Design of Decision Trees and Decision Diagrams*

Denis Popel, Elena Popel

State University of Informatics and Radioelectronics,
Laboratory of Image Processing and Pattern Recognition,
Brovky Str., 6, 220600, Minsk, Republic of Belarus,
E-mail: popel@bseu.minsk.by, popel_denis@yahoo.com

Abstract

This paper presents an approach to detect symmetries of switching functions for efficient design of Reed-Muller decision trees (DTs) and decision diagrams (DDs). The information theory measures of switching functions are used to determine possible symmetric variables and therefore to reduce search space. This approach allows to detect different types of symmetries of any variables coincident. We consider the technique to apply symmetries properties that can significantly improve Reed-Muller DT or DD design. We implement symmetry detection algorithm as a part of program for switching function minimization based on DT design. Experiments have been performed on MCNC benchmarks and the results verify the efficiency of our approach.

Key words: Switching functions, symmetry, decision trees and diagrams, information theory measures

1. Introduction

Determining symmetries among groups of variables is important in problems of logic synthesis [5], design verification and testing [7], and in problems of technology mapping, i.e. Boolean matching [10, 18]. The effectiveness of matching procedure can be increased if the groups of symmetric variables are known.

The symmetry properties are used in different areas of logic design. There are well known methods of circuit design, decomposition and minimization [4, 6, 15, 16]. In our investigation we focus on the detection of symmetries for Reed-Muller DT and DD design and further application for switching function minimization.

There are several techniques to recognize symmetries based on different principles, namely,

(i) manipulation of a truth table and truth column vector developed in this paper;
(ii) transformation of the given function into spectral domain;
(iii) formal representation of symmetric functions (decision trees and diagrams, Reed-Muller forms, etc.).

The well known algorithms explore properties of symmetries via manipulation of the truth tables. For example, in [17] an effective method to detect different types of symmetries based on numerical methods has been proposed.

The second direction exploits features of spectra to determine the symmetries in variables for given function. There are many results on detecting symmetries in Hadamard, Haar and other transform bases [5]. However, spectral coefficients are very expensive to compute and store for functions with large number of variables.

Formal representation of symmetric switching functions in positive polarity Reed-Muller (PPRM) expressions are studied for recognition symmetries in [1]. Authors use an additional program to obtain PPRM expression.

In recent years binary decision diagrams have been used as an efficient data structure to store

* This work was partially supported from Fund of Fundamental Researches (BELARUS).

functions, and symmetry detection has become feasible for functions with large number of variables [2, 8, 9]. Fast detection of symmetric variables is important for DTs and DDs design.

In our approach we consider switching function represented by truth tables or truth column vectors and use symmetric properties for efficient trees and diagrams design.

The paper is organized as follows. Section 2 outlines background of investigation and presents necessary definitions and notations. Section 3 presents an approach to detect different types of symmetries. Section 4 outlines algorithm to detect symmetries *InfoRECSym* and gives principles of Reed-Muller DTs and DDs design for symmetric switching functions. Section 5 presents experimental results and Section 6 concludes the paper.

2. BACKGROUND

Let give some essential definitions and prepositions that are important for the understanding the paper.

We use the following notations:

$X = \{x_1, x_2, ..., x_n\}$	a set of variables
$f(x_1, x_2, ..., x_n)$ or f	a switching function
f_x, $f_{\bar{x}}$	cofactors of f with respect to arbitrary variable x.
$\{x_i, x_j\}$	a *symmetric pair*
Ω	a set of expansion types
$H(f)$	*entropy* of switching function f
$H(f \mid x)$	*conditional entropy* of function f with respect to variable x

Switching function f can be represent as *positive Davio (pD) expansion*

$$f = f_{\bar{x}} \oplus x(f_{\bar{x}} \oplus f_x),$$

and *negative Davio (nD) expansion*

$$f = f_x \oplus \bar{x}(f_{\bar{x}} \oplus f_x).$$

We consider Reed-Muller DT or DD as directed acyclic graph. Each node is labeled with possible expansion ω with respect to arbitrary variable x. A couple (x, ω) is assigned to a node, where $x \in X$ and $\omega \in \Omega$. For Reed-Muller DTs and DDs: $\Omega = \{pD, nD\}$. *pD* and *nD* nodes and cofactors of expansion are shown in Fig. 1.

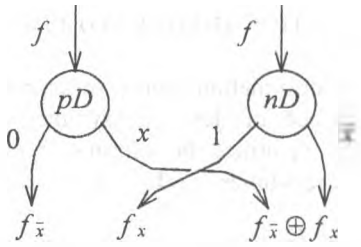


Fig. 1. Nodes and cofactors of *pD* and *nD* expansion of switching function *f* with respect to variable *x*.

2.1. SYMMETRIC SWITCHING FUNCTIONS

For any pair of variables x_i and x_j there are four cofactors $f_{x_i x_j}$, $f_{\bar{x}_i x_j}$, $f_{x_i \bar{x}_j}$, $f_{\bar{x}_i \bar{x}_j}$.

A function f is *nonequivalent symmetric* in variables x_i and x_j, denoted as $\{x_i, x_j\}$ or $\{\bar{x}_i, \bar{x}_j\}$, if f remains invariant when this variables are interchanged: $f_{\bar{x}_i x_j} = f_{x_i \bar{x}_j}$ [3].

A function f is *equivalent symmetric* in variables x_i and x_j, if it remains invariant when x_i and $\bar{x}_j$ ($\bar{x}_i$ and x_j) are interchanged: $f_{x_i x_j} = f_{\bar{x}_i \bar{x}_j}$ [3]. This type of symmetry is denoted by $\{x_i, \bar{x}_j\}$ or $\{\bar{x}_i, x_j\}$.

If function f is simultaneously nonequivalent and equivalent symmetric in x_i and x_j, then f is *multiform symmetric* [3].

Example 1.

(i) Function $f = x_2 \oplus x_3 \oplus x_2 x_3 \oplus x_1 \cdot x_2 x_3$ is nonequivalent symmetric in x_2 and x_3; function $f = x_1 \cdot x_3 + x_1 \cdot x_2$ is nonequivalent symmetric in x_2 and x_3;

(ii) $f = \bar{x}_1 \oplus x_2 \oplus \bar{x}_1 \cdot x_2 \cdot x_3$ is equivalent symmetric in variables $\{\bar{x}_1, x_2\}$ or $\{x_1, \bar{x}_2\}$;

(iii) $f = x_1 \cdot \bar{x}_2 + \bar{x}_1 \cdot x_2$ is multiform symmetric in $\{x_1, x_2\}$ ($\{\bar{x}_1, \bar{x}_2\}$) and $\{x_1, \bar{x}_2\}$ ($\{\bar{x}_1, x_2\}$).

A function f is *partially symmetric* with respect to $X_i \subset X$, if any permutation of variables in X_i leaves f unchanged.

A function f is *totally symmetric* if every pair of variables in the function is either nonequivalent or equivalent symmetric.

Example 2.

(i) Function $f = \bar{x}_1 \oplus x_2 \oplus \bar{x}_1 \cdot x_2 \cdot x_3$ is partially symmetric in variables $\{\bar{x}_1, x_2\}$ or $\{x_1, \bar{x}_2\}$.

(ii) $f = x_1 \cdot \bar{x}_2 + \bar{x}_1 \cdot x_2$ and $f = x_1 \cdot x_2 \oplus x_1 \cdot x_3 \oplus x_2 \cdot x_3$ are totally symmetric functions.

2.2. INFORMATION THEORY NOTATIONS

We use information theory measures to reduce the search space of detecting symmetric variables pairs. Besides we utilize the information measures for efficient Reed-Muller DTs and DDs design [11, 12, 14].

In order to quantify the information content revealed by the outcome for finite field of events with probabilities distribution, Shannon introduced the concept of entropy. *Entropy* of switching function f is given by [13]:

$$H(f) = -p_{|f=0} \cdot \log_2 p_{|f=0} - p_{|f=1} \cdot \log_2 p_{|f=1}. \quad (1)$$

We calculate probabilities $p_{|f=b} = k_{|f=b} / k$, where $k_{|f=b}$ is the number of assignments of values to variables (patterns) for which $f = b$, and k is the total number of assignments.

Example 3. Let us calculate entropy of switching function f given by truth column vector [1100000111000010]: $H(f) = -{}^{6}/_{16} \cdot \log_2 {}^{6}/_{16} - {}^{10}/_{16} \cdot \log_2 {}^{10}/_{16} = 0.95$ bit/pattern.

We consider the process of DT or DD design as recursive decomposition of switching function. A step of this recursive decomposition corresponds to the expansion of switching function f with respect to variable x. Assume that variable x of function f carries information that is, in some sense, the rate of influence of the input variable to output, f.

For positive Davio and negative Davio expansion we use conditional entropy $H^{\omega}(f \mid x)$ as information measure [14]:

$$H^{pD}(f \mid x) = p_{|x=0} \cdot H(f_{\bar{x}}) + p_{|x=1} \cdot H(f_{\bar{x}} \oplus f_x), \quad (2)$$

$$H^{nD}(f \mid x) = p_{|x=1} \cdot H(f_x) + p_{|x=0} \cdot H(f_{\bar{x}} \oplus f_x). \quad (3)$$

Thus, in proposed approach we consider the following tasks, that will be partially solved by information theoretic measures:

1. *How can we determine different types of symmetries in variables for a given switching function f?*
2. *How can we use symmetric properties for efficient design of Reed-Muller DTs and DDs?*

3. DETECTION OF SYMMETRIES

In this section we focus on detecting different types of symmetries (nonequivalent, equivalent, multiform, totally symmetry) by information measures, presented in subsection 2.2.

3.1. DETECTION OF NONEQUIVALENT SYMMETRY

Statement 1. Switching function f is nonequivalent symmetric in $\{ x_i, x_j \}$ or $\{ \bar{x}_i, \bar{x}_j \}$ if

$$f_{\bar{x}_i} \oplus f_{x_i} = f_{\bar{x}_j} \oplus f_{x_j} \quad \text{and} \quad f_{\bar{x}_i} = f_{\bar{x}_j}, f_{x_i} = f_{x_j}.$$

Proof. It is easy to show that $f_{x_i} = f_{x_i x_j} + f_{x_i \bar{x}_j}$ and $f_{\bar{x}_i} = f_{\bar{x}_i x_j} + f_{\bar{x}_i \bar{x}_j}$, hear '+' means union of cofactors. The nonequivalent symmetry condition $f_{\bar{x}_i x_j} = f_{x_i \bar{x}_j}$ implies $f_{x_i x_j} = f_{\bar{x}_i \bar{x}_j}$. Similarly, $f_{\bar{x}_i x_j} = f_{x_i \bar{x}_j}$. Hence, we can write $f_{x_i} \oplus f_{\bar{x}_i} = f_{x_j} \oplus f_{\bar{x}_j}$. ∎

DT or DD nodes with nonequivalent symmetric variables are assigned together as primitives (Fig.2).

Property 1. If switching function f is nonequivalent symmetric in x_i and x_j then

$$H^{pD}(f \mid x_i) = H^{pD}(f \mid x_j) \text{ and}$$
$$H^{nD}(f \mid x_i) = H^{nD}(f \mid x_j).$$

This property of entropy equality is necessary but not enough to detect symmetry.

Example 4. Consider switching function f of four variables that correspond to truth column vector [1100000111000010].

The cofactors for this function presented in Table 1. The information measures for pD and nD expansion are given in Table 2. According property 1 we analyze pairs with equal information measures. Nonequivalent symmetries in $\{ x_2, x_3 \}$ and in $\{ x_1, x_4 \}$ are possible.

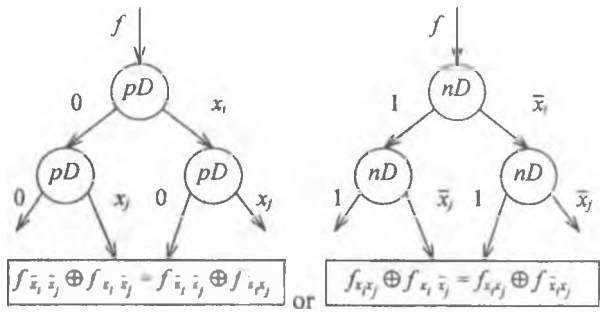


Fig. 2. Nodes of Reed-Muller DT or DD that correspond to nonequivalent symmetry.

Table 1. The cofactors to detect symmetries (Example 4).

	$f_{\bar{x}}$	f_x	$f_x \oplus f_{\bar{x}}$
x_1	[11000001]	[11000010]	[00000011]
x_2	[11001100]	[00010010]	[11011110]
x_3	[11001100]	[00010010]	[11011110]
x_4	[10001001]	[10011000]	[00010001]

Table 2. The information measures (in bit/pattern) for positive and negative Davio expansion (Example 4).

	$H^{pD}(f \mid x)$	$H^{nD}(f \mid x)$
x_1	*0.88*	*0.88*
x_2	0.91	**0.81**
x_3	0.91	**0.81**
x_4	*0.88*	*0.88*

The function f is nonequivalent symmetric in $\{x_2, x_3\}$: $f_{\bar{x}_2} \oplus f_{x_2} = f_{\bar{x}_3} \oplus f_{x_3}$ and $f_{\bar{x}_2} = f_{\bar{x}_3}$.

This function is nonequivalent symmetric in $\{x_1, x_4\}$: $f_{\bar{x}_1} \oplus f_{x_1} = f_{\bar{x}_4} \oplus f_{x_4}$ and $f_{\bar{x}_1} = f_{\bar{x}_4}$, taking into consideration the necessary permutation of variables assignments.

As a result of DT or DD design we obtain the following fixed polarity Reed-Muller expression: $f = 1 \oplus x_2 \oplus x_3 \oplus x_2 x_3 \oplus x_1 \cdot x_2 \cdot x_3 \oplus x_4 \cdot x_2 x_3$.

3.2. Detection of Equivalent Symmetry

Statement 2. Switching function f is equivalent symmetric in $\{x_i, \bar{x}_j\}$ or $\{\bar{x}_i, x_j\}$ if

$f_{\bar{x}_i} \oplus f_{x_i} = f_{\bar{x}_j} \oplus f_{x_j}$ and $f_{\bar{x}_i} = f_{x_j}, f_{x_i} = f_{\bar{x}_j}$.

Proof. It is easy to show that $f_{x_i} = f_{x_i x_j} + f_{x_i \bar{x}_j}$ and $f_{\bar{x}_j} = f_{\bar{x}_i \bar{x}_j} + f_{x_i \bar{x}_j}$, hear '+' means union of cofactors. The equivalent symmetry condition $f_{x_i x_j} = f_{\bar{x}_i \bar{x}_j}$ implies $f_{x_i x_j} = f_{\bar{x}_i \bar{x}_j}$. Similarly, $f_{\bar{x}_i \bar{x}_j} = f_{x_i x_j}$. Hence, we can write $f_{x_i} \oplus f_{\bar{x}_i} = f_{x_j} \oplus f_{\bar{x}_j}$. ∎

DT or DD nodes with equivalent symmetric variables are placed together (Fig.3).

Property 2. If switching function f is equivalent symmetric in x_i and x_j, then

$$H^{pD}(f \mid x_i) = H^{nD}(f \mid x_j) \text{ and}$$
$$H^{nD}(f \mid x_i) = H^{pD}(f \mid x_j).$$

This of entropy equality is necessary but not enough to detect symmetry.

Example 5. Consider switching function f of three variables that correspond to truth column vector [11100011].

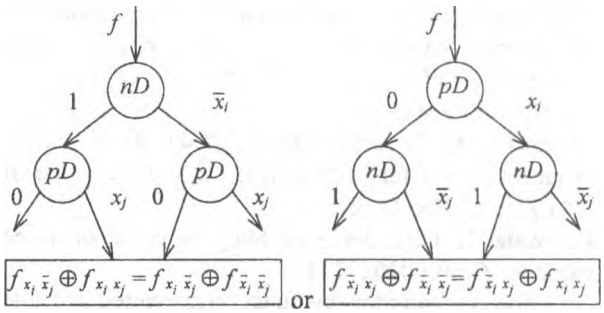


Fig. 3. Nodes of Reed-Muller DT or DD that correspond to equivalent symmetry.

The cofactors for this function given in Table 3. The information measures are presented in Table 5. Equivalent symmetries in $\{x_1, \bar{x}_2\}$ and in $\{x_2, \bar{x}_3\}$ are possible.

The function f is equivalent symmetric in $\{x_1, \bar{x}_2\}$: $f_{\bar{x}_1} \oplus f_{x_1} = f_{\bar{x}_2} \oplus f_{x_2}$ and $f_{\bar{x}_1} = f_{x_2}$ and $f_{x_1} = f_{\bar{x}_2}$.

This function is not equivalent symmetric in $\{x_2, \bar{x}_3\}$ cause $f_{\bar{x}_2} \oplus f_{x_2} \neq f_{\bar{x}_3} \oplus f_{x_3}$.

Finally, we obtain the following fixed polarity Reed-Muller expression: $f = \bar{x}_1 \oplus x_2 \oplus \bar{x}_1 \cdot x_2 \cdot x_3$.

3.3. Detection of Multiform and Totally Symmetries

Property 3. If switching function f is multiform symmetric in $\{x_i, x_j\}$, then

$$H^{pD}(f \mid x_i) = H^{pD}(f \mid x_j) =$$

Table 3. The cofactors to detect symmetries (Example 5).

	$f_{\bar{x}}$	f_x	$f_x \oplus f_{\bar{x}}$
x_1	[1110]	[0011]	[1101]
x_2	[1100]	[1011]	[0111]
x_3	[1101]	[1001]	[0100]

Table 4. The information measures (in bit/pattern) for positive and negative Davio expansion (Example 5).

	$H^{pD}(f \mid x)$	$H^{nD}(f \mid x)$
x_1	**0.81**	0.91
x_2	0.91	**0.81**
x_3	**0.81**	0.91

$$H^{nD}(f \mid x_i) = H^{nD}(f \mid x_j).$$

Example 6. (*Continue of Example 4*) The function f is multiform symmetric in $\{x_1, x_4\}$, cause $H^{pD}(f|x_i) = H^{pD}(f|x_j) = H^{nD}(f|x_i) = H^{nD}(f|x_j)$ = 0.88 bit/pattern.

Property 4. If switching function f is totally symmetric, then for each couple (x, ω) the value of $H^{\omega}(f|x)$ is the same.

Example 7. Consider switching function of three variables f = [00010111].

The cofactors for this function presented in Table 5. The information measures for *pD* and *nD* expansion are given in Table 6. Information measures for all variables are equal. Function f is totally symmetric: $f = x_1x_2 \oplus x_2x_3 \oplus x_3x_1$.

4. Algorithm to detect symmetries and principles of Reed-Muller DTs or DDs design

We propose an algorithm to detect symmetries of switching functions for Reed-Muller DT design. The algorithm called *InfoRECSym* (*Infor*mation *REC*ognizer *of Sym*metries) is described in Fig.4.

We incorporate the symmetry detection algorithm *InfoRECSym* to DT or DD design by following stages.

Stage 1. For each variable x of function f calculate information measures $H^{\omega}(f|x)$. Select subset of couples (x, ω) for which $H^{\omega}(f|x) \rightarrow \min$.

Stage 2. Check for symmetries according symmetry detection algorithm (Fig. 4).

Stage 3. For each symmetry pair construct DTs or DDs primitives (Fig.2 and Fig. 3).

Table 5. The cofactors to detect symmetries (Example 7).

	$f_{\bar{x}}$	f_x	$f_{\bar{x}} \oplus f_x$
x_1	[0001]	[0111]	[0110]
x_2	[0001]	[0111]	[0110]
x_3	[0001]	[0111]	[0110]

Table 6. The information measures (in bit/pattern) for positive and negative Davio expansion (Example 7).

	$H^{pD}(f\|x)$	$H^{nD}(f\|x)$
x_1	**0.91**	**0.91**
x_2	**0.91**	**0.91**
x_3	**0.91**	**0.91**

/* (input) $f = f(x_1, x_2, ..., x_n)$*/
/*(output) *symmetry pairs* $\{x_i, x_j\}$*/

InfoRECSym(f) {
  for each variable $x \in X$ {
    Determine cofactors f_0 and f_1 and store into *sub_table*:

pD	*nD*
$f_0 = f_{\bar{x}}$	$f_0 = f_x$
$f_1 = f_{\bar{x}} \oplus f_x$	$f_1 = f_{\bar{x}} \oplus f_x$

    Compute information measures $H^{\omega}(f|x)$ for *pD* and *nD* expansion, according (1)-(3)
  }
  for each pair $\{x_i, x_j\}$ {
    if $H^{pD}(f|x_i) = H^{pD}(f|x_j)$ and $H^{nD}(f|x_i) = H^{nD}(f|x_j)$
      if $f_{x_i} \oplus f_{\bar{x}_i} = f_{x_j} \oplus f_{\bar{x}_j}$ and $f_{x_i} = f_{x_j}$
      then f is nonequivalent symmetric in x_i and x_j
    if $H^{pD}(f|x_i) = H^{nD}(f|x_j)$ and $H^{nD}(f|x_i) = H^{pD}(f|x_j)$
      if $f_{x_i} \oplus f_{\bar{x}_i} = f_{x_j} \oplus f_{\bar{x}_j}$ and $f_{x_i} = f_{\bar{x}_j}$
      then f is equivalent symmetric in x_i and x_j
    if f is nonequivalent and equivalent symmetric simultaneously
    then f is multiform symmetric in x_i and x_j
  }
  if checked pairs are either nonequivalent or equivalent symmetric then f is totally symmetric function.
}

Fig. 4. Sketch of the algorithm to detect symmetries of switching function *f*.

Example 8. Let us consider the process of Reed-Muller DT design for switching function f that given by truth column vector [11111001]. The cofactors and information measures are presented in Table 7 and Table 8 respectively.

Step 1. According information measures for DT node we assign the couple (x_1, pD) (Fig. 5).

Table 7. The cofactors (Example 8, Step 1).

	$f_{\bar{x}}$	f_x	$f_{\bar{x}} \oplus f_x$
x_1	[1111]	[1001]	[0110]
x_2	[1110]	[1101]	[0011]
x_3	[1110]	[1101]	[0011]

Table 8. The information measures (in bit/pattern) for positive and negative Davio expansion (Example 8, Step 1).

	$H^{pD}(f \mid x)$	$H^{nD}(f \mid x)$
x_1	**0.5**	1
x_2	0.91	0.91
x_3	0.91	0.91

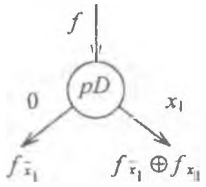


Fig. 5. Step 1 of DT design (Example 8).

Let us consider cofactor $f_{\bar{x}_1} = [1111]$. The cofactor is constant, therefore we assign a leaf (logic value 1) to Reed-Muller DT.

Step 2. Consider cofactor $f_{\bar{x}_1} \oplus f_{x_1} = [0110]$. The cofactors and information measures are given in Table 9 and Table 10 respectively. The information measures are equal. The cofactor $f_{\bar{x}_1} \oplus f_{x_1}$ is multiform symmetric in x_2 and x_3. Finally, we assign primitives with *pD* nodes for decision tree (Fig. 6).

We obtain the fixed polarity Reed-Muller expression: $f = 1 \oplus x_1 \cdot x_2 \oplus x_1 \cdot x_3$.

Table 9. The cofactors (Example 8, Step 2).

	$f_{\bar{x}}$	f_x	$f_{\bar{x}} \oplus f_x$
x_2	[01]	[10]	[11]
x_3	[01]	[10]	[11]

Table 10. The information measures (in bit/pattern) for positive and negative Davio expansion (Example 8, Step 2).

	$H^{pD}(f \mid x)$	$H^{nD}(f \mid x)$
x_2	**0.5**	**0.5**
x_3	**0.5**	**0.5**

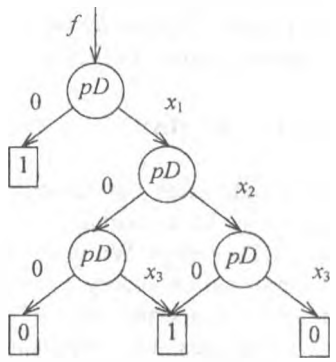


Fig. 6. Reed-Muller DT (Example 8).

5. EXPERIMENTAL RESULTS

We incorporate *InfoRECSym* algorithm in program of minimization of switching function via Reed-Muller DT design *InfoEXOR* [14] - on Pentium 100 MHz (RAM 48 Mb), programming language *C++* under OS Windows 95.

To verify the efficiency of symmetry detection approach, we tested program on MCNC benchmarks (completely specified Boolean functions). Table 7 contains fragments of our results. The column with label **in** shows the number of variables, column with label **out nr.** shows the output number (we consider single output benchmarks). In column *Time* the running times for the algorithms in CPU seconds are given. The column C_T refers the number of products in minimized FPRM expressions.

The running time of *InfoEXOR* program with

Table 11. Comparison of running times for Sym*pathy* [4], *InfoEXOR* [14] and *InfoEXOR* with algorithm *InfoRECSym*.

			Sym*pathy*	*InfoEXOR* (FPRM)	*InfoEXOR* (FPRM)+ *InfoRECSym*
	in/ out nr.	C_T	*Time*$^{\diamond}$, s	*Time*$^{\&}$, s	*Time*$^{\&}$, s
xor5	5/1	5	0.1	0.001	0.001
rd84	8/1	28	0.1	0.010	0.001
rd84	8/4	70	0.1	0.092	0.001
9sym	9/1	173	0.1	0.341	0.087
sym10	10/1	266	0.2	2.620	0.152
Total			0.6	3.064	**0.242**

-60%

13 times

$^{\diamond}$**Sparc 1+ workstation, OS UNIX**

$^{\&}$**Pentium 100MHz processor, OS Windows 95**

symmetry detection algorithm *InfoRECSym* is better for 60% than program Sym*pathy* [4] and in 13 times better than original program *InfoEXOR*.

6. Concluding remarks

This paper addresses the detection of different types of symmetries of switching function for Reed-Muller DTs or DDs design. We investigate symmetry detection by information theory point of view that gives us additional properties for switching function minimization techniques. Our program *InfoRECSym* successfully recognize symmetries for efficient Reed-Muller DTs or DDs design.

In future it will be interesting to extend our results to detect symmetries in multivalued functions.

References

[1] Butler J., Dueck G, Holowinski G., Shmerko V., Yanushkevich S., On the Recognition of Symmetries for Switching Functions in Reed-Muller Forms, *Int. Conf. on Pattern Recognition and Information Processing - PRIP'99,* vol. 1, 1999, pp. 215-234

[2] Butler J., Sasao T., On the Properties of Multiple-Valued Functions that are Symmetric in Both Variables and Labels, *IEEE Int. Symp. on Multiple-Valued Logic*, 1998, pp. 83-88

[3] Das S., Sheng C., On the Detecting Total or Partial Symmetry of Switching Functions, *IEEE Trans. on Computers,* vol. 20, no. 3, 1971, pp. 352-355

[4] Drechsler R., Becker B., Sym*pathy*: Fast Exact Minimization of Fixed Polarity Reed-Muller Expressions for Symmetric Functions, *IEEE Trans. on CAD of Integrated Circuits and Systems,* vol. 16, no. 1, 1997, pp. 1-5

[5] Edwards C., Hurst S., A Digital Synthesis Procedure Under Function Symmetries and Mapping Methods, *IEEE Trans. on Computers,* vol. C-27, no. 11, 1978, pp. 985-997

[6] Kim B., Dietmeyer D., Multilevel Logic Synthesis of Symmetric Switching Functions, *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 10, no. 4, 1991, pp. 436-446

[7] Liaw H., Tsaih J., Lin C., Efficient Automatic Diagnosis of Digital Circuits, *Int. Conf. on CAD,* 1990, pp. 464-467

[8] Miller D., Muranaka N., Multiple-Valued Decision Diagrams with Symmetric Variable Nodes, *IEEE Int. Symp. on Multiple-Valued Logic,* 1996, pp. 242-247

[9] Moller D., Mohnke J., Weber M., Detection of Symmetry of Boolean Functions Represented by ROBDD, *Int. Conf. on CAD,* 1993, pp.680-684

[10] Pomeranz I., Reddy S., On the Determining Symmetries in Inputs of Logic Circuits, *IEEE Trans. on CAD of Integrated Circuits and Systems,* vol. 13, no. 11, 1994, pp. 1428-1433

[11] Popel D., Fixed Polarity Reed-Muller Minimization of Incompletely Specified Boolean Functions Based on Information Estimations on Decision Trees, *Int. Conf. on Pattern Recognition and Information Processing - PRIP'99*, vol. 1, 1999, pp. 197-206

[12] Popel D., Cheushev V., Shmerko V., Yanushkevich S., Information Theoretic Approach for AND/EXOR Minimization, *Int. Workshop on Post-Binary Ultra-Large Scale Integration Systems - ULSI'99*, 1999, pp. 32-33

[13] Simovici D., V. Shmerko, V. Cheushev, and S. Yanushkevich, Information Estimation for Logic Functions, *Int. Conf. On Application of Computer Systems - ACS'97*, 1997, pp. 287-300

[14] Shmerko V., Popel D., Stankovic R., Cheushev V., Yanushkevich S., Information Theoretical Approach for Minimization of AND/EXOR Expressions of Switching Functions, *IEEE Int. Conf. on Telecommunications in Modern Satellite, Cable and Broadcasting Services -TELSIKS'99*, 1999 (Accepted)

[15] Suprun V., Fixed Polarity Reed-Muller Expressions of Symmetric Boolean Functions, *IFIP 10.5 Workshop on Application of the Reed-Muller Expansions in Circuit Design,* 1995, pp. 246-249

[16] Suprun V., Lichko Yu., A Method of Polynomial Expansion of Partially Symmetric Boolean Functions, *Int. Conf. on Computer-Aided Design of Discrete Devices - CADDD'97*, vol. 1, 1997, pp. 42-45

[17] Tsai C., Marek-Sadowska M., Generalized Reed-Muller Forms as a Tool to Detect Symmetries, *IEEE Trans. on Computers,* vol. C-45, no. 1, 1996, pp. 33-40

[18] Wang K., Hwang T., Boolean Matching for Incompletely Specified Functions, *IEEE Trans. on CAD of Integrated Circuits and Systems,* vol. 16, no. 2, 1997, pp. 160-168

A Neural Network for Combinatorial Problems of Optimization

Vladimir Golovko, S. Derechennik
Department of Computers and Mechanics,
Brest polytechnic institute, Moscowskaja 267,
224017 Brest, Republic of Belarus
cm@brpi.belpak.brest.by

Key Words: neural networks, optimization, NP-complete problems.

Abstract

A neural network architecture for the optimization problems is discussed. It is a feedforward neural network with fixed weights. Such approach consists in definition the weights, using a priori knowledge. This paper describes the neural network for solving optimization tasks.

1. Introduction

A lot of combinatorial tasks belong to the class of NP-complete problems. In this case optimum solutions can not be guaranteed to be found in polynomial time. Therefore a number of heuristics have been proposed to find approximate solutions [1,2]. Neural approaches based on different kinds of collective computation have gained interest [3-6]. This paper describes the neural network for combinatorial tasks. The architecture is inherently parallel and many units can carry out their computations at the same time. The principle of architecture and functionality of such network is shown on the example of solving the "shortest way" problem and knapsack problem.

2. The general architecture

The general architecture of the neural network is presented on Figure 1. It consists of 3 layers.

The input units receive data from outside the neural network and distribute these data to hidden units. The units in the hidden layer determine various variants of the decision of a task. A unit in the output layer is meant for definition of the optimum decision of a task:

$$Z_k=\text{opt}(y_j), \qquad (1)$$

where k – the number of the hidden unit, which identifies the optimum solution, Z_k – the optimum solution of a combinatorial task. Let's examine the decision of some optimization tasks.

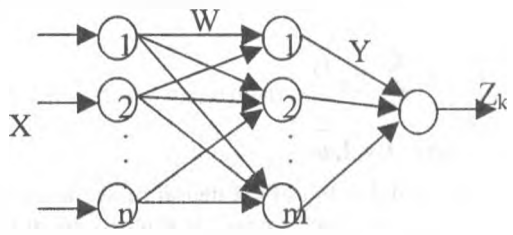


Figure 1. The general architecture of the neural network

3. The "shortest way" task

The "shortest way" problem consists in the following. Given n cities, initial and final points of route. The distances between various cities are known. What is the shortest way between the initial and final points of the route? Let's determine the structure of the neural network. The quantity of input units is given by the simple formula

$$p=(n-1)(n-2)+1, \qquad (2)$$

where n is number of cities.

The distance D_{ij} between the points of route i and j enters input units. It should be marked, that

$$i=\overline{1,n-1},\ j=\overline{1,n},\ i\neq j,\ j>i$$

for i=1.

So, if n=4, then p=7 and input vector of distances is given by

$$D=(D_{12},D_{13},D_{14},D_{23},D_{24},D_{32},D_{34}) \qquad (3)$$

Note that 1 characterizes initial and 4 – final points of route. The neurons in the hidden layer forms possible ways between initial and final points of the route. The quantity of the hidden units is as follows:

$$m = \sum_{i=1}^{n-1} \binom{n-2}{n-i-1} (n-i-1)! = (n-2)! \sum_{i=1}^{n-1} \frac{1}{(i-1)!} \quad (4)$$

where 0!=1.

It should be marked, that if n is very big, then we can obtain

$$\sum_{i=1}^{n-1} \frac{1}{(i-1)!} = e$$

In this case

$$m \approx e(n-2)!$$

Later on we shall represent the components of an input vector in linear system of coordinates

$$D = (D_1, D_2, ..., D_p) \quad (5)$$

Then the outputs of the hidden units is computed as

$$y_j = \sum_{i=1}^{p} w_{ij} D_i$$

where $j = \overline{1, m}$

The weights W_{ij} of the neural network are formed so as to receive a set of possible routes. The dimension of weight matrix is equaled pxm. So, if n=4, then m=5 and the weight matrix is as follows:

$$W = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix}$$

It is binary, if any ways between initial and final points of route are possible. In case, when there is no way between some points of route it is necessary instead of the 1 to put infinity (∞). For example, if there is no way between the first and second points of the route, i.e. D_{12}=Ø, then W_{11}=∞. Each column of the matrix W characterizes weight vector for the appropriate neuron. A unit in the output layer defines the shortest way

$$Z_k = \min_j \{y_j\}$$

where Z_k – the length of an optimum route; k – the number of an optimum route. If we know the number k of the winner, then the shortest way is possible to determine

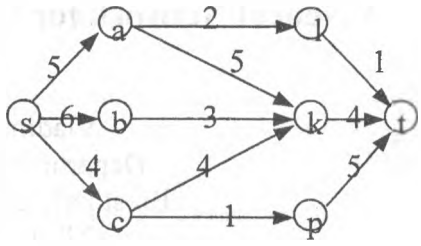


Figure 2. Source graph of ways

$$D_o = D^T W_k^T$$

where W_k – the weight vector of neuron k.

Example 1

Let a matrix distances between 4 cities be given. It is necessary to define the shortest way between 4 cities.

Tab. 8.1

	1	2	3	4
1	0	25	100	75
2	25	0	55	35
3	100	55	0	15
4	75	35	15	0

The input vector of distances is given by

D=(25, 100, 75, 55, 35, 55, 15).

Each neuron of the hidden layer defines one of the possible routes. Then

y_1=25+55+15=95

y_2=25+35=60

y_3=100+35+55=190

y_4=100+15=115

y_5=75

The output value of the neural network is

$$Z_k = \min_j \{y_j\} = 60$$

k=2

The weights of the second neuron (hidden layer) correspond to the following route:

$$D_o = \begin{bmatrix} D_{12} \\ D_{13} \\ D_{14} \\ D_{23} \\ D_{24} \\ D_{32} \\ D_{34} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} D_{12} \\ D_{24} \end{bmatrix}$$

As a result we can receive the optimum route:

$$1 \rightarrow 2 \rightarrow 4$$

The considered neural network is characterized by significant complexity. Therefore it can be used on final stages for search of optimum variant. Another way is the adaptation of a neural network to the algorithm of optimization, wich is used for the solution of the task. So, for example, if the dynamic programming is used for definition of the shortest way, then is easy to decrease the dimension of the neural network.

Let's consider the general approach of application of dynamic programming and neural network for the computation of the shortest way.

Let possible routes between an initial point S and final point T be given as the graph (Fig. 2). Such graph consist of four layers. Let's divide the task to steps and for definition of the optimum solution at each step we shall use the neural network. The first step coveres last three layers of the graph. Then it is necessary to find the optimum ways from points A, B and C to point T. For each of these tasks is formed the neural network. The number of cities is 4. Therefore the structure of the neural network will be the same, as in example 1. As a result of performance of the first step, we can receive

$$Z_2(a) = 3;\ a \rightarrow l \rightarrow t,$$

$$Z_1(b) = 7;\ b \rightarrow k \rightarrow t,$$

$$Z_4(c) = 6;\ c \rightarrow p \rightarrow t$$

Let's present the graph as follows (Fig. 3).

The neural network for n=5 is used for the definition of the optimum route from S to T. In this case

$Z_7(S) = 8$.

and route will be the following:

$$s \rightarrow a \rightarrow l \rightarrow t$$

If only real routes are taken into account, then the

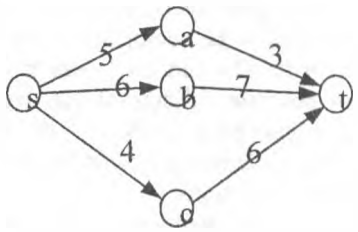


Figure 3. Reformed graph of ways

dimension of the hidden layer decreases. So, for n=5 the number of hidden neurons is

m = 10

If the real graph is taken into account (Fig. 2), then

m=3.

Thus, the use of the dynamic programming permits to decrease the dimension of the neural network.

4. The knapsack problem

The knapsack problem is NP-complete and is formulated as follows: (given a set of subjects u = (u_1, u_2, ..u_n) and for each of them volume V(u) and cost C(u) are known. It is required to fill in the bag of limited volume T so, that to obtain maximum cost of the packed things or to make it more than K. The mathematical formulation of the knapsack problem can be presented as follows:

$$\sum_u V(u) \le T \quad \max_u \left\{ \sum_u C(u) \right\},$$

And

$$\sum_u V(u) \le T \quad \sum_u C(u) \ge K,$$

And

were N is a given cost. The general architecture of the neural network is shown on Fig 4.

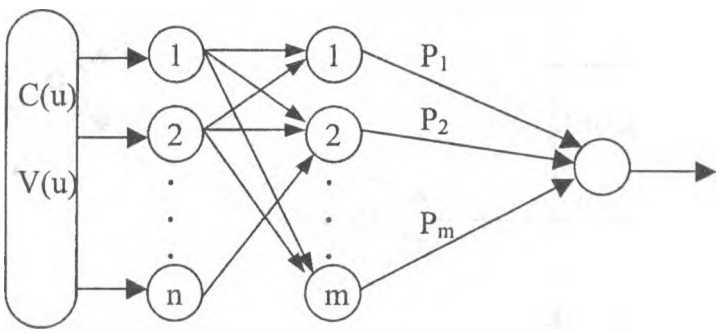


Figure 4. Neural network for solving the knapsack problem

It consist of three layers. The input information of the neural network is the vector of the cost C(u) and of the volume V(u). The hidden layers is intended for definition of possible variants of the decision of the knapsack problem. Each neuron of the hidden layer consists of three elementary neurons (Fig 5), which carry out various functions.

The element b_i generates i-th variant of the decision of the task. For this purpose the following function is calculated:

$$K_{bi} = \sum_{j=1}^{n} w_{ji} C(u_j) \tag{6}$$

The element d_i is the neural element with threshold function of activation. It analysis i-th variant of the decision as follows:

$$S_i = \sum_{j=1}^{n} w_{ji} V(u_j) - T \tag{7}$$

$$K_{ai} = \begin{cases} 1, & if\ S_i \le 0 \\ 0, & otherwise \end{cases} \tag{8}$$

The element P_i performs the following function:

$$P_i = \begin{cases} K_{bi}, & if\ K_{ai} = 1 \\ 0, & otherwise \end{cases} \tag{9}$$

The output layer consists of one unit, which defines the optimum variant of the decision:

$$Z_k = \max_i \{P_i\} \tag{10}$$

where Z_k – the maximum cost of the subjects in the bag; k – the number of the unit, which identifies the optimum variant of the knapsack problem.

Using number k it is possible to define the final decision:

$$U = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \begin{bmatrix} w_{1k} \\ w_{2k} \\ \vdots \\ w_{mk} \end{bmatrix} \tag{11}$$

The number of the hidden units depends on the technology of the decision knapsack problem. Generally, if all variants of the decision are analyzed, then the hidden layer contains the following quantity of neural elements:

$$m = 2^n - 1. \tag{12}$$

Then the weights vector of i-th neuron corresponds to binary code of the number i: so, if i=5 and n=6

$W_5 = (00101)$

The weight matrix is formed so, that each column was able to characterize the weights of the certain neural element. So for n=3 and m=7 weight matrix is defined as follows:

$$W = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{13}$$

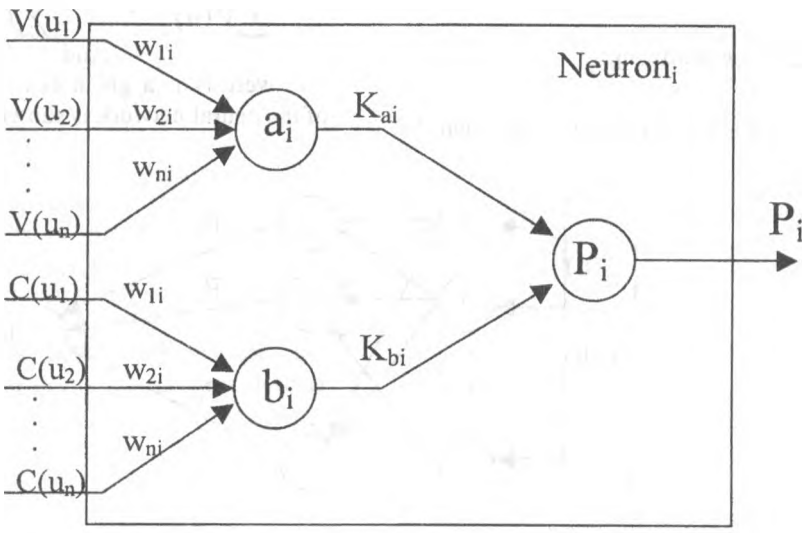


Figure 5. Scheme of neural element

The disadvantage of the above mentioned neural network is a great number of hidden units. For elimination of this disadvantage the algorithm of optimization to the neural network can be adapted, as it was in the previous section. Another way is the increase of the tacts of the work of the network. Suppose, there is a neural network consisting of 2^n neural elements. It is necessary to solve on such a neural network the task, which has dimension r, where r>n. Let

m=r-n

and the neural network, which contains r input units and 2^n-1 hidden units is given.

For the decision of knapsack problem (size r>n) on such a network it is necessary to spend 2^m tacts. The weight of neural elements will be changed in each tact according to the weight matrix.

Example 2

Let's consider the example of the decision of knapsack problem of dimension n=3. Let

$$V(k) = \{5,7,10\},$$
$$C(k) = \{3,2,1\},$$
$$T=17.$$

The quantity of neurons of the hidden layer is

$$m = 2^3 - 1 = 7$$

The weight matrix is determined according to expression(13). Then the results of calculations of neural elements of the hidden layer can be presented in the table 2.

Tab.2

№	Ka	Kb	P
1	1	1	1
2	1	2	2
3	1	3	3
4	1	3	3
5	1	4	4
6	1	5	5
7	0	6	0

Let's define the output value of the neural network:

$$Z_k = \max_i \{P_i\} = 5$$

$$k = 6.$$

According to number k of a neuron and its weight vector W_k we identify the decision of the knapsack problem:

$$U = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

Thus optimum decision is achieved by packing in bag subjects u1 and u2.

5. Conclusion

In this paper the neural network for the optimization problems is described. It consists of 3 layers and allow to find global optimum. The weight matrix of the neural network is computed, using a priori knowledge. Such approach can be used for different combinatorial problems of optimization.

Literature

1. Lin, S.&B.W.Kernighan, "An effective heuristic for the travelling-salesman problem," *Optimization Research* 21(1973), 498-516.
2. Padberg, M.W.&G.Rinaldi, "Optimization of a 532-City Symmetric Travelling Salesman Problem by Branch and Cut," *Operations Research Letters* 6(1987), 1-7.
3. Burr, D.J. "An Improved Elastic Net Method for the Travelling Salesman Problem," in *IEEE International Conference on Neural Networks*#1, IEEE, New York, 1988, 69-76.
4. Hopfield, J.J.and Tank, D.W. "Collective computation with continuos variable." *Disordered systems and Biological organization* (Springer-Verlag), 1986, 155-170.
5. Melamed, J.J. "Neural network and combinatorial optimization," in *Proceedings of the 16th IFIP Conf. Syst. Model Optim.* (Compieqne, France), 1993, 537-542.
6. V. Golovko, H. Suhodolsky, V. Dimakov. "Neural Net for Combinatorial Optimization" *Proc. of intern. conf. "High Performance Computing"* (april, Boston) San Diego: The Computer Simulation International, - 1998.

Fuzzy Expert System for a Stock Trading Process

R. Simutis, G.Stankevicius, A.Veckys
Process Control Department, Kaunas University of Technology,
Studentu 48, LT-3031 Kaunas, Lithuania,

Abstract

The goal of this study was to build and to evaluate a human skill based fuzzy expert system for a decision making support in stock trading process. Our focus was concentrated on computer software that is capable to reproduce the knowledge from the skilled stock trader. Using classical technique and soft computing methods the expert system STRASS (Stock TRAding Support System) was developed. The proposed system was tested for the historical collection of NASDAQ, NYSE and AMAX stock records. At present, it is being tested by "KOLEGU" mutual fund in a real stock trading process.
Key words –fuzzy expert systems, stock trading process, evolutionary programming

1.Introduction

The financial and economic applications of soft computing methods have attracted a lot of interest in past years. It had become clear to many financial market observers that soft computing and computer science tools, especially those from the fields of fuzzy systems and neurocomputing were frequently finding relevance in the financial markets [1]. In the stock markets, the relationships between process variables are generally too complex to make grounded decisions using classical system theory.

The goal of this study was to build and evaluate a human skill based decision support system for a stock trading process using classical and soft computing methods. The essence of soft computing is that unlike the traditional, hard computing, it is aimed at an accommodation with the imprecision of the real world, like stock trading process. Thus the guiding principle of soft computing is: exploit the tolerance for imprecision, uncertainty, and partial truth to achieve tractability, robustness, low-solution cost, and better rapport with reality [2]. Soft computing is based on fuzzy logic methods, neurocomputing, evolutionary programming and parts of machine-learning theory [3,4].

For the decision support system development we have used so called "hybrid system approach". The most important stock's financial indicators were used to form the data basis for the decision support system and the information fusion and decision-making algorithm was developed using fuzzy technique and knowledge from experienced stock trading experts. Then the evolutionary programming methods were used for the further improvement of the fuzzy expert system.

The data transfer and preprocessing algorithms, also realization of fuzzy algorithms and membership function tuning algorithms were merged to the software packet ***STRASS*** (**S**tock **TRA**ding **S**upport **S**ystem, Windovs95, MATLAB 5.2). The proposed expert system yielded about 26 % annual "paper profit" for the historical collection of NASDAQ, NYSE and AMAX stocks records. At present, it is being tested by ***"KOLEGU"*** mutual fund in a real stock trading process.

2. Elements of decision support system

2.1 Formation of data basis

The data basis for the decision support system contains information on selected NYSE, NASDAQ and AMEX stocks. The information was obtained using *StockQuest* software (http://www.marketguide.com). By selection of stocks the special rules were applied:

The first critical rule for stock selection was the annual growth rate of earnings per share *(EPS)* over the last 3 years. Earnings per share were calculated by dividing the company's total after tax profits by the company's number of common shares outstanding. Only the stocks with the *EPS* growth rate over 50% was picked out for the next selection step. *StockQuest* search engine was used during processing this rule. About 600 stock-candidates from 9200 companies were selected for the next exploration step.

In the second step the multi-linear regression between input variables (rolling *EPS*, growth rate of *EPS*) and output variable (stock price) was identified (data records

from the last three years). Then the "fair" price of the stock was estimated using this regression model. The stock was selected for the next exploration step, if the "fair " price was at minimum 10% larger that the actual price.

In the third step the recommendations of Wall Street's Experts were taken into account (http://quote.yahoo.com/). Only the stocks with the quality index QI<3.0 (range 1-5, strong buy=1,strong sell=5) was picked out for the next selection step.

During the forth selection step the following formal constraints were applied for analyzed stocks: Price/Earning ratio < 100, Price>10 $, Average Daily Volume >15000. At the end of this selection procedure about 70-80 stocks were selected for the expert system data basis. This selection procedure has to be repeated every month.

Additionally, the data basis was extended using data on recent general stock market direction and actual 'stock price moving' direction. For the characterization of general market direction the S&P500 index was used. Last 30 S&P index values were fitted using linear regression technique and slope of this curve was used for the characterization of general stock market direction. Actual 'stock price moving' direction was estimated using stock price for last five trading days. The relative stock price values were fitted using linear regression technique and this slope value was used then as a fuzzy expert system input.

2.2 Linguistic variables and terms

Experienced stock trading experts are able to keep the trading process in such a manner that their profits are significantly better than those of other traders. In this study our focus was concentrated at first on computer software that is capable to extract the knowledge from a skilled stock trader. An expert knowledge about the stock trading process was formulated in terms of fuzzy variables and rules and mounted into the fuzzy expert system. We use the following linguistic variables to describe decision support system inputs and output.

Input variables:

- Price Error Factor, PEF, (fair/actual price ratio), with descriptive terms BIG (B), MEDIUM (M), SMALL (S).
- Expert Opinion, EO, with terms STRONG BUY (SB), BUY (B), NEUTRAL (NE).
- General Market Direction, GMD, with terms POSITIVE (P), NEUTRAL (NE), NEGATIVE (NG).
- Stock Price Moving direction, SPM, with terms POSITIVE (P), NEUTRAL (NE), NEGATIVE (NG).

The output variable of the expert system is the Buy Sell Recommendation, BSR, with terms STRONG BUY (SB), BUY (B), HOLD (H), SELL (S) and STRONG SELL (SS).

2.3 Fuzzy rules and inference mechanism

The knowledge from the skilled stock trader was converted to a set of fuzzy rules, which can be regarded and processed with computer. Thus, the starting point of this construction is a representation of process input/output relationships by mean of fuzzy 'IF - THEN '-rules, formulated by stock trading expert. The *l-th* of *M*-rules may be written as follows:

$R^{(l)}$: **IF** x_1 is F_1^l **AND** ..., **AND** x_n is F_n^l
THEN y is G^l,

where F_i^l are the fuzzy sets (terms) of the fuzzy input vector $x = [x_1, x_2, ..., x_n]$. We consider rule systems with a single fuzzy output variable y only (buy-sell recommendations), its terms are denoted by G^l. To define the input/output membership functions we used Gaussian bell-shaped function [5]. It should be mentioned that the membership functions are normalized to a unit maximum. In order to process the set of fuzzy rules an appropriate inference mechanism and a defuzzification technique must be chosen. In the case of the learning procedure to be applied to the system of rules, the product inference rule proved to be convenient [5]. As an improved defuzzification technique, the 'modified center average technique' has been applied. The knowledge from the skilled stock trader was represented by 81 rules, which have the following form:

IF *PEF* is *BIG* **AND** *EO* is *SB* **AND** *GMD* is *P* **AND** *SPM* is *P* **THEN** *BSR* is *SB*.

2.4 STRASS software

The proposed decision support system was realized on PC/Windovs95 using MATLAB 5.2 and Personal Stock Monitor as a basic software. (http://www.personaltools.com/psm/). The data transfer and preprocessing algorithms, also realization of fuzzy algorithms and membership function tuning algorithms were merged to the software packet ***STRASS*** (Stock **TRA**ding **S**upport **S**ystem). The packet has a user friendly GUI and can be used by people from financial institution, that haven't special education in programming languages.

3. Application of decision support system

The key steps by real application of expert systems are as follows:

- define quasi on-line all input variables for fuzzy expert system using *INTERNET* services, ***STRASS*** software and stock market data,
- determine the output of fuzzy expert system for every selected stock, using preprogrammed inference mechanism and defuzzification technique,
- arrange all stocks according to buy-sell recommendation quantitative value [1,-1],
- if the buy-sell signal value exceeds some threshold value (e.g. >0.7), make recommendation for stock owning,
- if the buy-sell signal value is smaller as some threshold value (e.g. < -0.5) make recommendation for stock selling.

4. Results and conclusions

The quality of the expert system was tested by "paper profit making " process for selected stocks using 6 month historical data records (01.01.1996-30.06.1996) and 10000 $ start capital. The start capital was divided in four equal parts and four different stocks were traded during this time interval. The original expert system was capable to make about 18 % annual profit on start capital (price differences minus commissions, commissions: 10$ pro trade, such a service is provided by e.g. DATEK company, http://www.datek.com). For the improvement of the original decision support system (parameter of the membership functions of linguistic variables) evolutionary programming technique was applied [3,5]. The improved decision support system yielded about 26 % annual "paper profit" for the selected stocks, for the time interval, that was not used in evolutionary programming procedure.

At present, the developed fuzzy expert system is being tested by ***"KOLEGU"*** mutual fund in a real stock trading process. The fund has achieved about 24.5 % annual profit in the year 1998.

5. References

1. Proceedings of the IEEE/IAFE computational intelligence for financial engineering, IEEE Service Center, New Jersey, 1997.
2. Tsoukalas, L.H., Uhrig, R. E. (1997) Fuzzy and neural approaches in engineering, John Wiley & Sons, Inc, New York.
3. Fogel, D.J. (1995) Evolutionary computation: toward a new philosophy of machine intelligence, IEEE Press, New York.
4. Wang, L.X. (1994) Adaptive fuzzy systems and control, Prentice Hall, Englewood Cliffs.
5. Simutis, R., Lübbert, A. A comparative study on random search algorithms for biotechnical process optimization, J. Biotechnology, 1997, 52,245-256.

THE APPLICATION OF SELF-ORGANIZING NEURAL NETWORKS IN FINANCIAL MARKET ANALYSIS

Giedrius Stankevicius
Kaunas University of Technology, Studentų 48-327, Kaunas, LT-3028, Lithuania
phone: 370-7-798886, e-mail: giedrius@ earthling.net

ABSTRACT

The problem of comparison of different companies is facing, when analyzing company's performance in stock exchange market. Due to many different financial ratios and parameters sometimes it is almost impossible to decide which company is a leader or not. One of the ways to solve this problem is the use of self-organizing (Kohonen's) neural networks. Using financial parameters as inputs, as an output we will have different groups of companies. Using the ranking, which is made before, results it is possible to determine which group consists of leading companies. By adding financial parameters of concrete company to the existing network, therefore, company will appear in one of earlier formed groups. Now it is possible to decide about mentioned company's price changing tendencies in the nearest future.

Key words: *self-organizing neural networks, clustering, stock market.*

1. Introduction

Due to the globalization of financial markets, expansion of the electronic trade and the growth of information about the market, the specialists of investment funds more frequently try to use artificial intelligence methods for the market analysis [2]. These methods are widely used in so-called intelligent process control and monitoring systems [5]. Works related with the use of neural networks, obscure sets methods and expert systems for financial analysis and formation of trade decisions, receive the greatest recognition and interest in financial markets. The possibilities of self-organizing networks for clustering shares of different companies and formation of "stock leaders" groups are analyzed in this work. Discussed methods are presenting as one of the possible recommendations for the investing companies.

Stock analysts more frequently try to combine the methods of technical and financial analysis lately. The growth of quantity of the parameters characterizing the company's shares and the information that is necessary to work up is enormous in this case. Some methods that are able to work fast with the information and are able to select the most decisive parameters when we have in mind the tendencies of the share price change are proposed in this study.

2. Factors that influence the stock market

Many factors decide the share price change in the market. One of the main problems of stock market analysis is to establish the relations between these factors, the share price and the tendencies of share price change. This task is very complicated because of complexity of the relations between factors.

Before the explanation of these relations, it is necessary to overview those factors that, according the opinion of stock market analysts, have the most influence on share price change. The factors can be divided in three groups: objective factors, speculative factors and subjective factors. Objective factors can be divided into micro and macro level factors. [4]

Micro factors have an effect on the concrete company's share price. Company's financial state, the size of the company, net profit margin, the perspectives of the industry sector, to which company belongs, are known as micro factors. Besides above-mentioned factors, very specific factors of company's policy have an influence the share price changes.

Macro factors have an effect on the group of the companies or the whole market. Some of the macro factors are: the stability of economical system, the level of saving and the size of State debt, conjuncture of gold, real estate and commodities, economical growth rates, inflation, international money flow rates, the state of currency system, etc.

Expectations are speculative factor. Sometimes expectations have more influence on the market than objective factors. Then the most important thing is the possibilities to win because of differences of share price rate. The dividends and interest rates are not so important in this case.

The significance of subjective factors in the stock markets is very big. By their nature, they can occur because of technical stock market analysis, prediction methods and because of the opinion of

financial analysts. When we are talking about financial market analysis, it is necessary to mention, that analysts can different interpret the same fact.

It is difficult to estimate all mentioned parameters in quantities. Sometimes it is impossible. Thus, it is impossible to decide how these factors relate. Technical and fundamental analysis methods and parameters with clear meanings are used for the financial market analysis.

Some certain premises should be taken into account, when using technical analysis methods. It is accepted that price changing tendencies could be established with the use of prehistoric data of stock parameters and factors [3]. There are some of the parameters, which are used for the technical stock analysis:

1) shares outstanding;
2) opening price of share;
3) closing or last bid price of share;
4) highest price of share during the trading day;
5) lowest price of share during the trading day;
6) average daily trading volume;
7) the rate of different stock indexes, etc.

The detailed analysis of the company activities and its financial reports is carrying out, when using fundamental stock analysis. Stock market analysts turn their attention at these parameters [1]:

1) earning per share growth;
2) return on assets and equity;
3) debt to assets and equity;
4) net profit margin, etc.

As it was mentioned earlier, stock analysts more frequently try to combine the methods of technical and financial analysis lately. The growth of quantity of the parameters characterizing the company's shares and the information that is necessary to work up is enormous in this case. Some methods that are able to work fast with the information and are able to select the most decisive parameters, when we have in mind the tendencies of the share price change, are proposed in this study.

3 The application of neural networks for the market analysis

The use of neural networks is based on the possibilities of neural networks to estimate the relations between different market factors during the training. The merit there is that relations between different factors are not fixed a priori, but are estimated during the training with the help of experimental data. Thus their outputs are safe from so-called 'human factor', when we obtain the desirable result, but not the result which shows what is hiding inside the data. The attempt to find the relations between the share price and different market factors during the network training is made. The technical and financial data about the market's state and the company's financial state is used as the inputs for the neural network in this case. When sufficient quantity of data is presented for the network training, it is possible to identify complex non-linear dependencies between analyzing variables and share price. It is also possible to find the tendencies of share price change. Unfortunately, different subjective factors take strong effect on the stock market and neural network is incapable to estimate them. Thus, functional dependencies, which are identified after the training of network, should be estimated very carefully. The reliability of these dependencies drastically depends on the quantity and the quality of owned data.

3. Self-organizing neural networks. Basics and training.

The major function of self-organizing networks is to automatically classify input patterns into a number of disjoint clusters. The patterns located in the same cluster have similar features. The self-organizing networks is formed in terms of unsupervised learning, i.e. learning without a teacher, for instance winner-take-all competitive learning. Here we introduce the algorithm of competitive learning self-organizing networks. In a self-organizing network, a vector quantizer can be performed by adjusting weights from N input nodes to M output nodes. When the input patterns have been presented sequentially to the network without specifying the desired output, the input patterns can be automatically classified into M clusters. The structure of self-organizing neural networks and the geometrical explanation of competitive learning are schematically illustrated in **Figures 1** and **2**.

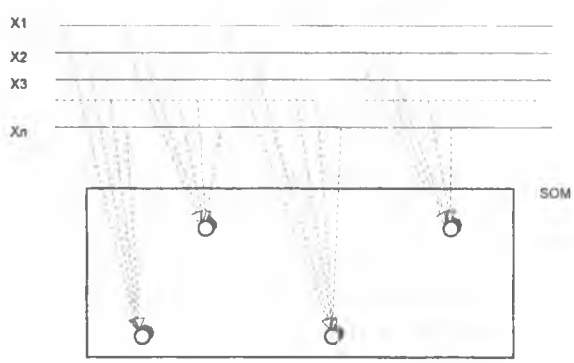


Figure 1. Topology of Kohonen self-organizing network

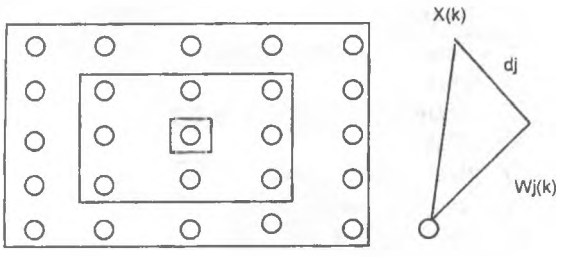


Figure 2. Geometrical illustration of competitive learning

The detailed learning algorithm can be summarized as follows:

Step 1: Randomly initialize small valued weights

$$W_{ij}(0),\ i=1,\ N;\ j=1,M.$$

Step 2: Present input vector:

$$x_i,\ i=1,N.$$

Step 3: Calculate the distance between the input vector and the weight vector for all individual output nodes:

$$d_j = \sum_{i=1}^{N} (x_i(k) - W_{ij}(k))^2 \qquad (1)$$

Where $x_i(k)$ is the input to the node i at time k, and $W_{ij}(k)$ is the weight from input node i to output node j at time k.

Step 4: Select the most active output node j^*, or the so-called 'winner' which has the least distance, i.e.

$$d_{min} = \min\{d_j, j=1,M\} \qquad (2)$$

If $d_j=d_{min}$ then $j=j^*$ and $y_j^*=1$, otherwise $y_j=0$ (j is not equal j^*).

Step 5: Upgrade the weights for the 'winner' node

$$W_{ij}(k+1)=W_{ij}(k) + g(k)\, y_i\, (x_i-W_{ij}(k)) \qquad (3)$$

Where $g(k)$ denotes learning rate and is defined as a time decreasing function within the range (0,1).

Step 6: Repeat by going to Step 2.

From equation (2), we can see that eventually the weights are upgraded only for the winner node j^*. However, in practice, the weights can also be upgraded only for the winner node j^*, but also for all nodes in the neighborhood of the winner. The size of the neighborhood $NE_j(k)$ can be predefined and can start large and slowly decrease in size with time. The weights upgrading may follow a modified version:

$$W_{ij}(k+1)=W_{ij}(k) + g(k)\, (x_i-W_{ij}(k)) \qquad (4)$$

for all j which are located in the neighborhood $NE_j(k)$.

There are many different software packages for the realization of self-organizing networks.

In our study, we will use a demo version of **Viscovery® SOMine** software (http://www.eudaptics.com/index.html).

4. Preparation of the data

For the testing of possibilities of self-organizing neural networks for financial market analysis, the data from USA stock exchanges was used. The first step was the screening of the companies. We used **StockQuest®**, software for this operation. This convenient software is available through the Internet. The user has a possibility to build up a portfolio from about 50 different factors. The screening is implementing, when certain factors were chosen. The first problem is to pick up empirically 12-15 factors, by recommendations of financial market analysts, and use these factors to build up a portfolio, which consists of potential to buy shares. Second problem is to rank these selected shares. To solve this problem certain weights were given to the factors. Weights were given according the opinion of financial analysts [1]. Third, and the main problem, is to divide selected shares in some groups (leaders, losers) with the use of self-organizing neural networks.

5. Clustering course and the results.

Because of limited size of the article, the solving of the first and the second problems are overviewed shortly. The realization of the third problem is analyzed more extensively. As it was mentioned above, at first, 15 factors and their meanings were chosen according to the opinion of the financial analysts. Using the **StockQuest** software, the selection of companies was made. The portfolio consisted of 30 companies. Then certain meanings of weights were empirically given to the factors, which characterized the companies. Right after this, the ranking of the companies was made - by estimation of all weights, company was assigned to its place in the general listing. Companies situated from leaders to the losers. During the next step, it was necessary to make sure, that companies can be divided into clusters corresponding to the general listing. We used **"Viscovery SOMine"** software for the clustering operation. There are some restrictions in the demo version of the software. The amount of the factors available to analyze is nine. Because of this reason, it was impossible to make clustering operation with the whole amount of factors at once.

The parameters were changed one with each other. This restriction has merit - after detailed analysis it is possible to decide which factors are decisive when dividing the companies in the groups. A few factors were declined to make easier the changing the factors one with each other. It was factors with absolute meanings, because they depend on the size of the company, the quantity of the workers in the company, etc. The parameters with relative meanings were left for the further analysis and clustering. After the first attempt of clustering, it was impossible to determine any group of companies, neither leaders nor losers (according the ranking list) (**Fig. 3**). The numbers on the figure mean the lace of the company in the ranking list.

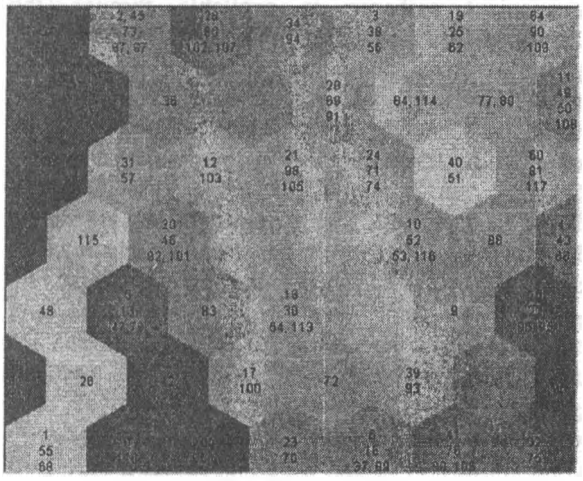


Figure 3. Clustering result after the first attempt

Then clustering vas made with different combinations of the factors. Thus, the influence of different parameters was analyzed. The influence of separate factors was analyzed too (there is the special option in the software). After the detailed analysis it became clear, which factors have the most effect on the formation of groups (**Table no. 1**)

Table 1. Recommended set of factors for the screening and clustering of the companies

No.	Factor	Meaning
1	Earning per share growth rate, 3 Years (%)	>25
2	52 week price percent change (%)	>25
3	Current price of share, ($)	>10
4	Revenue growth rate, 3 years (%)	>25
5	Return on average common equity, TTM (%)	>15
6	Long term debt to total equity, quarterly	<1
7	Insider ownership percent (%)	>20
8	Institutional percent owned, (%)	>5
9	Net profit margin, TTM (%)	>5
10	Price to earnings ratio, excluding extraordinary items, TTM	<100

After the clustering operation with the factors from the **Table 1**, the results showed that few groups of the companies were formed (**Fig. 4**). The group of the losers appeared most clearly. Other companies scattered chaotically in the upper left and the lower right corners. These companies were not shown for the clarity of the picture.

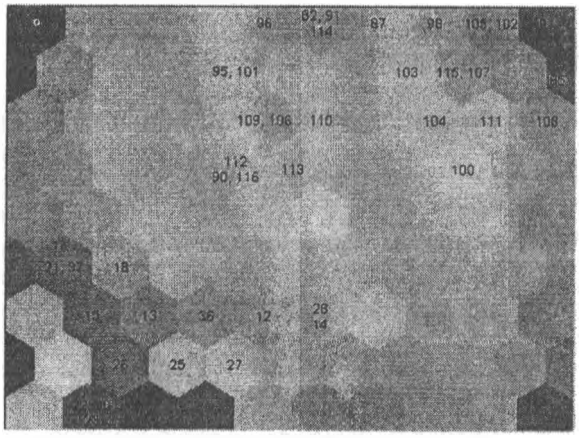


Figure 4. Clustering result with the selected set of factors

Finally, the result was not satisfying enough. The groups didn't appear clearly, some losers were in the same clusters with the leaders. The problem is that the ranking weights were selected subjectively. Thus, the ranking list is also subjective. The results of ranking and clustering varied. Adopting the weights would have an effect on the ranking results, but it can lead to the non-objective results.

By adding financial parameters of concrete company to the existing network, therefore, company will appear in one of earlier formed groups. Now it is possible to decide about mentioned company's price changing tendencies in the nearest future.

The following algorithm is the summarized clustering process algorithm:

1) Different companies are selected by the recommendations of financial market analysts. Recommended set of factors for selection (screening) is shown on the **Table 1**.

2) The ranking list of selected companies should be built. The weights for the ranking list are selected according the opinion of financial analysts.
3) All selected companies should be clustered.
4) By comparing the ranking list and the clustering results, groups of leader and loser companies should be determined.
5) The data about the company of our interest should be selected. The parameters of the data should be the same as for the companies selected earlier.
6) This data should be added to already trained self-organizing neural network. The company should appear in one of the earlier formed groups. Now it is possible to decide about mentioned company's price changing tendencies in the nearest future.

6.Conclusions

Following conclusions was made according the results of the study:

- Self-organizing neural networks can be used as the powerful tool for the financial market analysis.
- Selected parameters are usable (with some exceptions) for the clustering of the companies.
- The result of the clustering is the formed groups of leading and losing companies.
- Clustering can be used for creation of the decision support systems for the financial market analysis.

REFERENCES

Allrich, T. (1995). *The on-line investor. How to find the best stocks using your computer*, 240 p. St. Martin's Griffin, New York.

Marshall, J., Ed. (1997). *Conference on Computational Intelligence for Financial Engineering (CIFEr)*, 307p., Crowne Plaza Manhattan, New York.

Perk, R. (1990). *Professionelle Aktienanalyse fur jedermann*, 135 p. Verlag C.H.Beck, Munchen.

Rasteniene, A. (1998). *Vertybinių popierių rinka*, 64p., Lietuvos informacijos institutas, Vilnius.

Tsoukalas, L., Uhrig, R. (1997). *Fuzzy and neural approaches in engineering*, 585 p., John Wiley & Sons, New York.

Neuroelectronic Systems: Binding Neurons to Electric Circuits.

Molchanov P.G. Denisov A.A. Martinovich G.G. Cherenkevich S.N.

Abstract

This paper is devoted to description of a new type of information system based on interaction of biological neurons and electronic components - neuroelectronic (NE) system. Main features and components, like neural network sensor, of NE are briefly put into consideration followed with an example of electric interaction between neural network of a snail Lymnaea Stagnalis neurons and electronic MOSFET sensor.

Introduction

Living brain and traditional computer have been always considered as very different information systems. Nevertheless, having completely different information structure biological neural networks and electronic computers operate with trains of electrical signals while coding information on the basic structural level. This a fortunate similarity resulted nowadays in numerous efforts of constructing some kind of hybrid system with direct electric coupling of neurons and electronic components - neuroelectronic (NE) system, a system aimed at elimination of a wide difference gap between living neural networks and artificial computers [1].

In order to build NE system it is necessary to establish direct electrical interaction of alive nervous cells included in a neuron network and artificial component - sensor. It is important to take into account the fact, that interaction should be provided simultaneously in many (hundred, thousand) points of a neural network on a enough long period of time. Traditional electrophysiological methods operating for these purposes with glass microelectrodes cannot be considered acceptable because of complexity of manipulation and size of a construction. Besides, any operation with glass microelectrodes causes the cell death because of membrane rupture.

At the same time, on the basis of micro technologies it is possible to construct an array of flat microtransducers providing strong contact to cell bodies [2]. The recording of neurons electrical activity with the aid of macroelectrode array was carried out for the different cell systems. It was shown that the application of the similar electrode structure allows to detect cooperative and individual electrical activity of neurons in culture of a nervous cells [3]. However, major number of electrodes requires utilization of massive external equipment of an amplification and commutation. Besides, rather large distance between a source of a signal and amplifier arises the big parasitic capacitance and low signal to noise ratio which makes an interpretation of obtained data for a microelectrode system very complex.

On the other hand, modern microelectronic technologies allow creation of an external sensors with the array of active elements (usually field-effect transistors) built-in exactly in the cell placement area [4-5]. Such active sensors have a small distance between a source of a signal and amplifier that open new possibilities for research information processing in neurons.

If made such microtransduces array in the form of a specialized chip with neurons placed in a physiological solution directly on its surface one can name it the neurochip - sensor for neuron networks or neurosensor.

Cell culture for neurosystem.

One of the main element of any NE system are nervous cells. Their behavior in artificial conditions, keeping the property to generate biopotentials, ability of differentiation and saving vital activity during long-lived time (about one month and more) in conditions in vitro determines operation range of the NE system.

Now it is much known about neuron cell. The mechanisms of generation of rest, operation and synaptic potentials, mechanism of spike propagation on a cell and its neurites, morphological structure of a cell and etc. However the following level - interaction of cells and cell-like associations - is researched very poor.

For a creation of NE systems it is important to clarify what is the main functioning unit of the nervous network: one synapse, collection of synapses, site of a membrane, cell or other structures. In this respect culture of neurons creates exclusive possibilities for detection of elementary functioning unities of the nervous system: logical elements, signal converters, memory and integrators.

In culture in pretty wide limits it is possible to change composition of a culture medium, chemical composition of a substrate, add or remove from medium any physiologically active substances with the influence on spike activity and synapses.

The cell of a nervous tissue in conditions of culture in general save the metabolic characteristics of a cell of the whole organism. The values of activity of an oxidizing metabolism, exchange of enzymes, mediators, nucleic acids and protein, ionic carrier often can be comparable to values obtained in situ.

The sensor of electrical activity.

The second main element of NE system is the array of microtransdusers which are capable to record external electrical signals of nervous cells. One main feature of neurons electrical signals is their low power, that is caused by specificity of cells as biological objects. The maximum value of voltage and current amplitude constitute accordingly about 100 mV and 10^{-8} A at a resistance of a cell-like membrane about 10 MOm. It impose a number of the requirements external sensors of cell electrical activity should satisfy.

One of a main requirements, which arises because of high resistivity of a cell membrane, is the necessity to use primary transducers with a high input resistance. Most approaching for this purpose are the field-effect transistors, which have an input resistance about 1 GOm.

For insolation of sensor units from a solution the photoresistive polymer - polyimide is usually used. However thin polyimide film is not stable in a saline solution. This does not allow to carry out long-term experiments without disruption of coating integrity followed with sensor death. Therefore we designed other method based on construction peculiarities of microelectronic component in glass-to-metal frame.

The transistor consists of a silicon chip packed in a steel frame with outputs in glass insulators. The connection of contact sites of a chip (gate, drain, source) with outputs is performed through golden wire with a diameter about 30 microns. For an access to a chip the top of package was cut off, then the wire connecting a contact site of a gate and appropriate output, was detached from output and was drawn up. Three prepared transistors were placed side by side with a distance between gate wires was minimum and then were flooded by polymer. After solidification of polymer, a top was cut forming a flat site with three gold micro contacts. For carrying out of measurements the sensor was built into bottom of the culture camera.

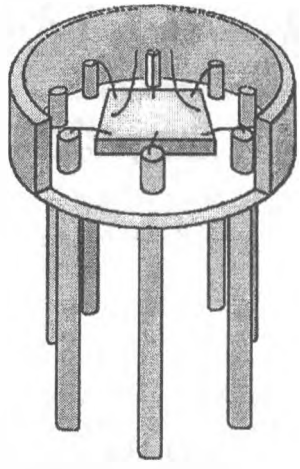


Fig. 1. A construction of the microelectrode sensor on the basis of the field-effect transisor.

Described method allows to create insulating coatings within the width of 2 mm., that provides reliable protection of a chip against salts of a solution without usage of photopolymers. The golden wire as electrodes has chemical resistance and biological inertness necessary for carrying out of long-term experiments.

Registration of electrical signals of neurons.

Detected external signals of neuron usually have a high level of inphase interferences. Therefore, to reject interference, system of an amplification should connect transducers in differential manner.

In an input cascade of the amplifier the differential circuit of field effect transistors was used. One transistor achieves signal from the neuron another from a reference electrode. The transistors were switched on in the saturation mode, therefore the drain-sources currents of transistors were modulated by the voltage on the gates. The drain voltage was fed to a differential amplifier, which magnified a signal,

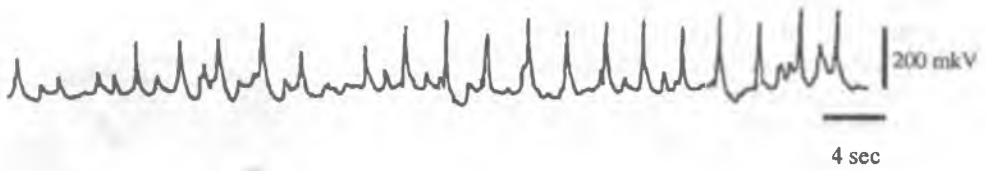


Fig. 2. Action potentials of a neuron on a background of synaptic potentials registered with application of sensors based on MOSFET.

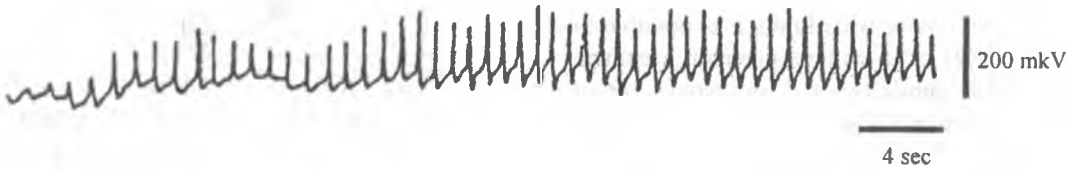


Fig. 3. A spike train of a neuron registered with application of sensors based on MOSFET.

rejecting a zero signal current and inphase interferences on the inputs.

The given technique was successfully applied for long-term monitoring of electrophysiological activity of neurons of a snail Lymnaea stagnalis.

For culturing of a nervous tissue of a snail Lymnaea stagnalis the following media composition was used: 30 % L-15 medium, glucose - 400 mg/ml, antibiotics (penicillin - 100un/ml, streptomycin - 50un/ml), 70 % of a saline solution of the following composition: NaCl 3 g/L; MgCl 0,14 g/L; KCl 0,127 g/L; CaCl 0,455 g/L. The isolated ganglions of snail nervous system were cultured within several weeks. During this time the electrophysiological activity of a neurons was recorded. The detection of activity was carried out on the basis of above described sensor, which was embedded in the bottom of neurons culture chamber.

An electrical activity of snail nervous tissue recorded with the help external sensor represented in a fig. 2-3.

References

1. Bove M., Grattarola M., Tedesco M. and Verreschi G. (1994). Characterization of growth and electrical-activity of nerve-cells cultured on microelectronic substrates - towards hybrid neuro-electronic devices. - Journal Of Materials Science Materials In Medicine V.5, P: 684-687.

2. Bove M., Grattarola M., Martinoia S., Verreschi G. (1995). Interfacing cultured neurons to planar substrate microelectrodes - characterization of the neuron-to-microelectrode junction. - Bioelectrochemistry and Bioenergetics V.38, P: 255-265.

3. Breckenridge L. J., Wilson R. J. A., Connolly P., Curtis A. S. G., Dow J. A. T., Blackshaw S. E., Wilkinson C. D. W. (1995). Advantages of using microfabricated extracellular electrodes for in-vitro neuronal recording. - J. Neurosci. Res. V.42, P: 266-276.

4. Hammerle H., Egert U., Mohr A., Nisch W. (1994). Extracellular recording in neuronal networks with substrate integrated microelectrode arrays. - Biosensors & Bioelectronics V.9, P: 691-696.

5. Janossy V., Toth A., Bodocs L., Imrik P., Madarasz E., Gyevai A. (1990). Multielectrode culture chamber: a device for long-term recording of bioelectric activities in vitro. - Acta Biol Hung V.41, P: 309-320.

APPLICATION of GENETIC ALGORITHMS FOR SOLUTIONS OF THE TASK IS FREQUENT - TERRITORIAL PLANNINGS GROUP RADIO ELECTRONIC EQUIPMENT

Kulaga V.V., Muravjev S.A., Spiridonov S.V., Telyatnikov R.V.
Republic Belarus, Military academy RB, ph.: 269-48-82

Abstract - One from the important problems, is a problem of limited frequency's number between radio electronic equipment's (REE) within large group. It is necessary that all REE may operate with specific correlation of signal-noise (interference) on boundaries of their service's zones, con sideling of electromagnetic compatibility. The aim of the work is the process of recommendations on application of genetic algorithms (GA) for solution of the optimal forming of frequencies - territorial scale of REE.

Key words - electromagnetic compatibility, genetic algorithms.

The high efficiency of REE stimulates fast rates of their distribution in all areas of human activity. According to this the electromagnetic situation becomes moke difficult, consequently arises the problem of electromagnetic compatibility in condition frequencies's channel limitation. One from such problems, is the problem is distribution frequency channel in limited condition, within a large group of REE, with a proper correlation of signal-noise at the boundaries of their operational zones. The solution of given problem amount to optimization is frequent - territorial scale (FTS). At a moment this problem is decided by consideration review of several pairs, with the consequent complex processing results of researches. For example, a radio set is selected with which a individual assigned frequency. Then the pair variants of operation this REE with other one are examine, next the most appropriate frequencies for these of REE of classification are selected. This procedure is repeated a lot of times for all classification. As a result each REE has a specific frequencies, but in case that the radios received frequency in last turn the station in which the last REE will has not appropriate level of signal-noise on boundary of operating zone.

The aim of this publication is processing of the recommendations on application of GA for the solution of frequency-territorial distribution for all REE classification's.

The simple GA, includes three simples operators, there are the following:

- Operator of reproduction (OR)
- Operator of crossing (OC)
- Operator of mutation (OM)

OR is know as the artificial version of natural selection. The OR realizes a choice best of chromosomes for the next operation of GA, it is operation of crossing. OC is a casual rupture chromosomes of the same size with their consequent connection in a new casual combination. This fact supports the search of an optimum on all area of acceptable solutions. The OG realizes "moving" on total area of solutions using the "jump". It allows to quit from local optimum, but also detection of global extremum does not guarantee. The mutation is a casual genetic changes within chromosome, were intended for finding news qualitative properties, intrinsic defined solution.

The GA provide a random search of suitable solutions, the way of optimization fitness function (FF). In a role of FF for given problem, it's to use value describing a level a signal - noise on a boundaries of operating zones all REE in classification. The value of genets can accept values of all allowed frequencies in classification. After casual creation of a source population (set chromosomes) the OR selects chromosome for OC and OM, which create qualitatively new by casual image. If the values FF of these solutions a rather great, they can get in a new population and to participate in the next cycles of operation of GA. This the population chromosomes with good values of FF is created, from this values (as the best) chromosomes are selected (for given problem it is a best FTS).

Accuracy and rate of finding of the best solution depend on parameters GA, manner of coding information in chromosomes, also from the size of an initial population.

The number of experiments was conducted. The aim of it was detection of degree of influence of varied parameters GA on tate and accuracy of solution's finding of given above problem. These parameters were the various procedures of choice chromosomes

on reproduction stage, level and degree of mutation in strings (lines), also the number of chromosomes in an initial population.

The experiments have shown, that the it is more chromosomes in initial population, then the faster GA finds appropriateble solution. It's a fact that calculating difficult is increasing and there fore process of operation GA becomes more durablis in time. It'is necessary to find the conciliatory proposal. According to the authors opinion, such solution is the size of initial population approximately equal (0.25 — 0.30) *n chromosomes, where n — amount of researched objects (for our case, n the number REE in classification).

In reproductions stage the procedure of chromosome's choice renders significant influence to a course of calculating. Were investigated a simples, biquadratic, relative roulette and procedure of cut of the worst solutions. In usual roulette game, the ball has absolute identical chances to stay in anyone sectors. But if the roulette is divided into unequal sectors (fig.1), the probability roulettes ball will stay in broad sector, is great, and narrow — is small.

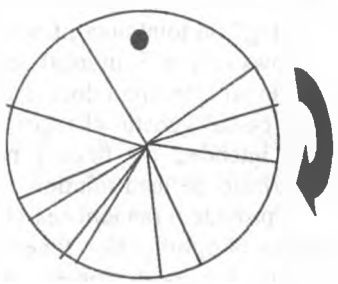


Fig. 1

With the reference to function of GA, each sector of formed simple roulette corresponds to one chromosomes and the size of the sector is proportional to this chromosome's goal function. The inequality of roulette sectors leads to nonuniformity of chromosomes choice. The casual character of choice can not lead in choice chromosomes with extremely good value FF, but for average OC and OM will be selected the best chromosomes, which will participate in creation of acceptable solution.

The main difference between biquadratic and simple roulette is, that FF of each of chromosomes is raised in the fourth degree. It provides increasing of probability of a choice chromosomes with large value of FF. For example: let there are two chromosomes in population the values of these chromosomes are equal 0.4 and 0.6. The probabilities of choice these chromosomes using simple roulette are equal 0.4 and 0.6 accordingly, and using biquadratic are equal 0.16 and 0.84. The wide difference is observing on initial stages of operation GA, then the FF chromosomes have small values. On final stage, when FF chromosomes are rather great and are identical, biquadratic roulette operates as simples.

The operation of relative roulette considerably differs from simple one. The operation of relative roulette can be described as follows: the worst line must be selected from all population, it value of FF is multiplied on 0.95. The obtained result takes as a start of counting. Next recalculating FF from each of chromosomes of population in relative values is further produced. These values are calculated as difference between the value of FF each chromosomes and starting of counting. The is "stretching" range of values FF chromosomes of population is produced. The total value of all relative values FF chromosomes of population undertakes square of all slabe of roulette, and sectors of formed roulette are the relative values from each chromosomes. It is a fact that the best chromosomes in relative roulette have probability of a choice for the next procedure more than ten times greatly than worse ones. Let's return to an offered example: the worse lines or (chromosomes) with FF equal 0.4 is multiplied on 0.95. As a result we have value equal 0.38, this number must be received as a beginning of counting. The relative value of worst chromosomes is equal 0.4-0.38 = 0.02, and for best ones 0.6-0.38 = 0.22. Therefore probability of choice for best chromosomes 0.22 / (0.22 + 0.02) = 0.92, and worse ones is - 0.08. Utilization of relative roulette gives advantages before idle time on initial stages of operation GA. In a final phase of algorithm operation the application of such means is undesirable, as the probability of choice chromosomes using relative roulette does not depend on values of them FF. The relative registres residual between the best and worst chromosomes only, and for OC and OM will be selected same chromosomes. It provides narrowing of searching space of suitable solution.

OR operation "cut" deletes from populations the worst solutions. This procedure perfectly operates, when the researched function has "smooth" sort fig.2a, or it has some equivalent extremums fig.2b. In case that researched function has sort shown at fig.2c, the application of the given procedure very seldom leads to finding sought solution, thus in case of hitting in one of local extremums "cut" leads in fast convergence of population. OC here does not operate. It does not allow to leave from deep local extremum.

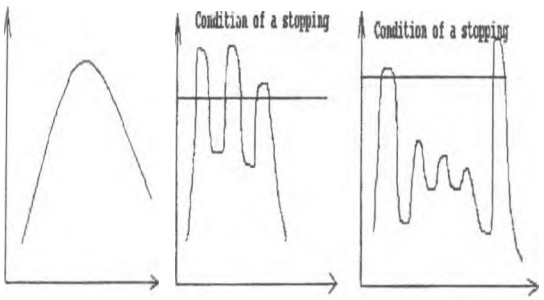


Fig. 2a Fig.2b Fig.2c

This or that procedure uses depending on stage of calculations. In the beginning of calculations, when FF has small values, the relative and biquadratic roulette "operates" perfectly, just these procedures provides " most directed " search at the expense of selection only best chromosomes, that on initial stage is positive feature. At this stage "direction" of search is selected and the eventual result depends on its selection. On final a stage, when the founded solution differs insignificantly from the value of stopping criterion of GA, it is recommended to use a simples roulette, which in difference from relative, allows to select not only best chromosomes populations, but also chromosomes with a small value FF. It provides the extension of searches area of acceptable solution. The fig.3 is represented the dependence's between finding of acceptable solution and the number of steps of GA's operation with a diverse procedures of selection chromosomes at stage of reproduction.

If during function GA has got in a local extremum, it is recommended to use "biquadratic" roulette, with high level of mutation, or depositing in population of so-called "chaos". In a role of "chaos" it is recommended to introduce to population accidentally generated chromosomes.

The leaded researches also have shown, that according to amount of steps GA and intermediate results of FF it is possible to judge about the degree of "variability" of researched function surface. For example, if GA during small number of operation cycles

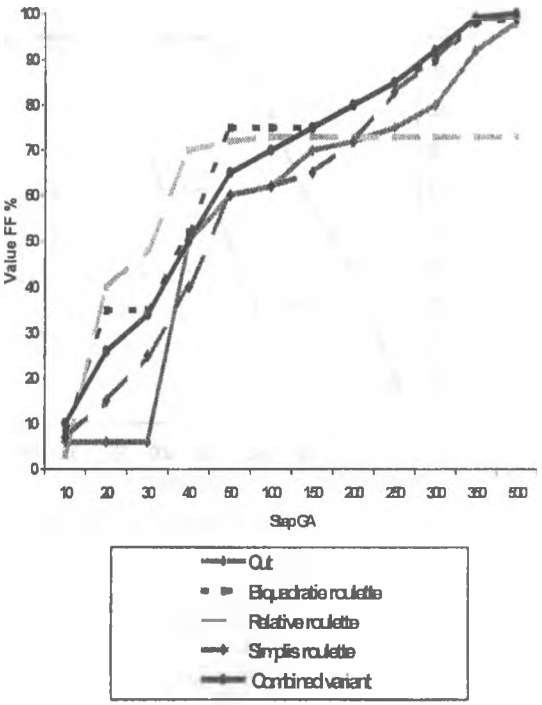


Fig. 3

finds suitable solution and the procedure of "cutting" is used, that researched function has "smooth" form with exactly expressed by one or several identical local extremums, satisfying to creation of stopping. If GA finds the suitable solution with using biquadratic or simples roulette only, and at that values of FF best of chromosomes populations are stationary for a long time, it is possible to draw a conclusion, that the researched function has "ravine" kind. The fig.4 has presented two graphic, which permitting to judge about degree of "variability" of the researched function. One of them is corresponded to the "smooth" function, and another one to "ravine" kind.

The recognition of the researched function allows to give the recommendations at the choice of parameters GA for providing solution of the concrete optimization problem.

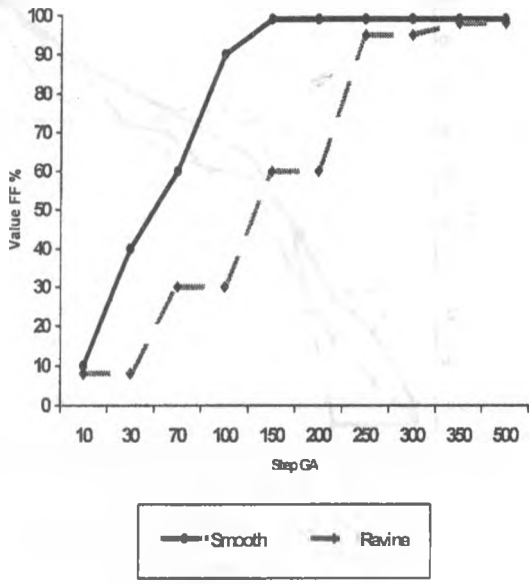


Fig. 4

In the conclusion it would be desirable to notice, that GA can use successfully for other problem of integral optimization, such as the problem of direct-sales, representative of the problem of tracing and packing, but each new problem requires the unique customizations.

The literature

Апорович А.Ф. Статистическая теория электромагнитной совместимости радиоэлектронных средств. Минск: Наука и техника. 1984г.

Architecture of Computer Vision Software System

A.Ivkin, A.Doudkin, R.Sadykhov

Institute of Engineering Cybernetics of the National Academy of Sciences of Belarus
6 Surganov Str., Minsk, 220012, Belarus
sadyhov@newman.basnet.minsk.by, doudkin@newman.bas-net.by

Abstract

The software of the computer vision systems is complicated. The authors discuss the principles of computer vision systems organization and propose the architecture for such software system.

Introduction

This article describes the architecture for high performance software system that supports distributed computing and parallel processing. Vision is one of the most difficult abilities to be implemented artificially. Developments in this area often require knowledge from a vast range of areas, including artificial intelligence, mathematics, physics, computer architectures and neurosciences.

Computer vision rose unique challenges in computer software and hardware technologies. The challenges include making computer vision systems less expensive, more compact and more robust in various conditions. These systems often have critical missions and require real-time processing of data to meet millisecond-level control cycle requirements. A comprehensive approach to developing, operating and maintaining a computer vision system should cover all these critical issues[1]:

- a scalable, open architecture;
- a comprehensive, computer-aided rapid development and deployment environment;
- real-time control and operator-interaction capability.

The ultimate goal is to propose a unified architecture solution for the computer vision software. Scraping existing systems and building new architecture from scratch is not practical. So, computer vision technologies must evolute incrementally. First, let's look at fundamental computer vision technologies.

1. Function chart of the computer vision system

The common function chart of the computer vision system is given in fig. 1. First, the image of an object is transferred to the light – signal converter through the optical device, then the electrical signal is amplified and is stored in the pre-processing device. The image analysis (secondary processing) device carries out location and recognition of the object, determination of its coordinates and orientation. The information on the object is displayed by the device of the visual control in case of need. On the basis of the received information the communication controller chooses control signals, that bring in action executive mechanisms, carrying out influence on the object. Besides the system of computer vision can carry out record of the results of the analysis of the image. E.g., it can generate the description of the electronic circuit on the basis of the processed image of the VLSI chip.

The purpose of the primary processing device is to reduce the common time of the image processing. In the time of pre-processing representation of the image is done as a set of characteristic points (the centre of the object, its important characteristics: corners, distances from centre up to edge, etc.) Thus, at the first stage direct processing and representation of the information are done in form that is convenient for further standard transformations which usually are carried out on the universal computer.

The secondary processing device carries out recognition of the image, i.e. it uses its characteristic attributes for determination, to which class the object belongs. On the input of the device there is a code of the image, obtained at the stage of the primary processing. On the output there are results of the processing of the image as the description in a

language of the subject area. In the case, when the system is intended for automated control, on the output we have control orders, translated by the controller of communication in signals, determining movings and actions of the manipulator.

The important part of system is the control device. Its functions are control of the processing devices parameters and synchronization of the processes, carried out in the system.

2. Further development of the computer vision system architecture

For further specification of the classes structure of the computer vision system we list some basic methods of image processing and recognition used in similar systems [2] (fig. 2).

It is necessary to mean, that the process of image input and generation and processing is affected by noise. A goal of pre-processing is elimination of noise distortions, brought in at different stages of formation, i.e. restoration of true values of the brightness function [3].

This purpose is pursued by image filtering. We shall consider the basic filtering methods:

1. threshold;
2. anisotropic and reccurent;
3. discrete Fourier transform, and also Hadamard and Walsh transforms that are less exact, but faster.

At the following stage of processing the image segmentation is carried out [4]. The segmentation is a separation of the image into components: objects, their fragments and characteristics. Two approaches to the decision of a problem of segmentation are known.

The first one is based on contour location on the images, that is assumed as a set of borders between object and background, between various objects or between adjacent areas of the same object. Three basic methods of contour location of the image are known:

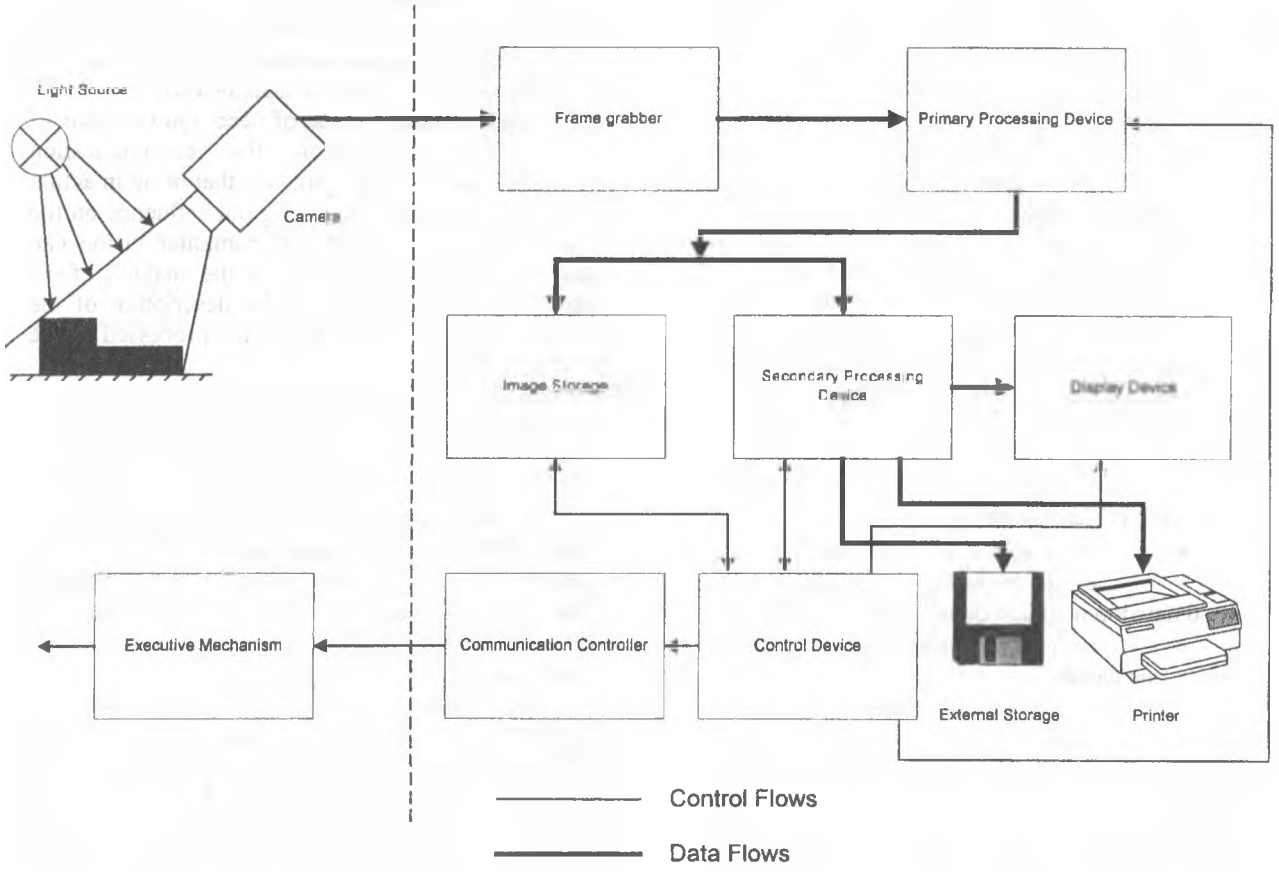


Fig.1. Function Chart of the Computer Vision System

1. method of spatial differentiation;
2. method of functional aproximation;
3. method of high-frequency filtering.

The specified methods of contour location of the image are realized in the computer differently. The realization of a method of spatial filtering brings

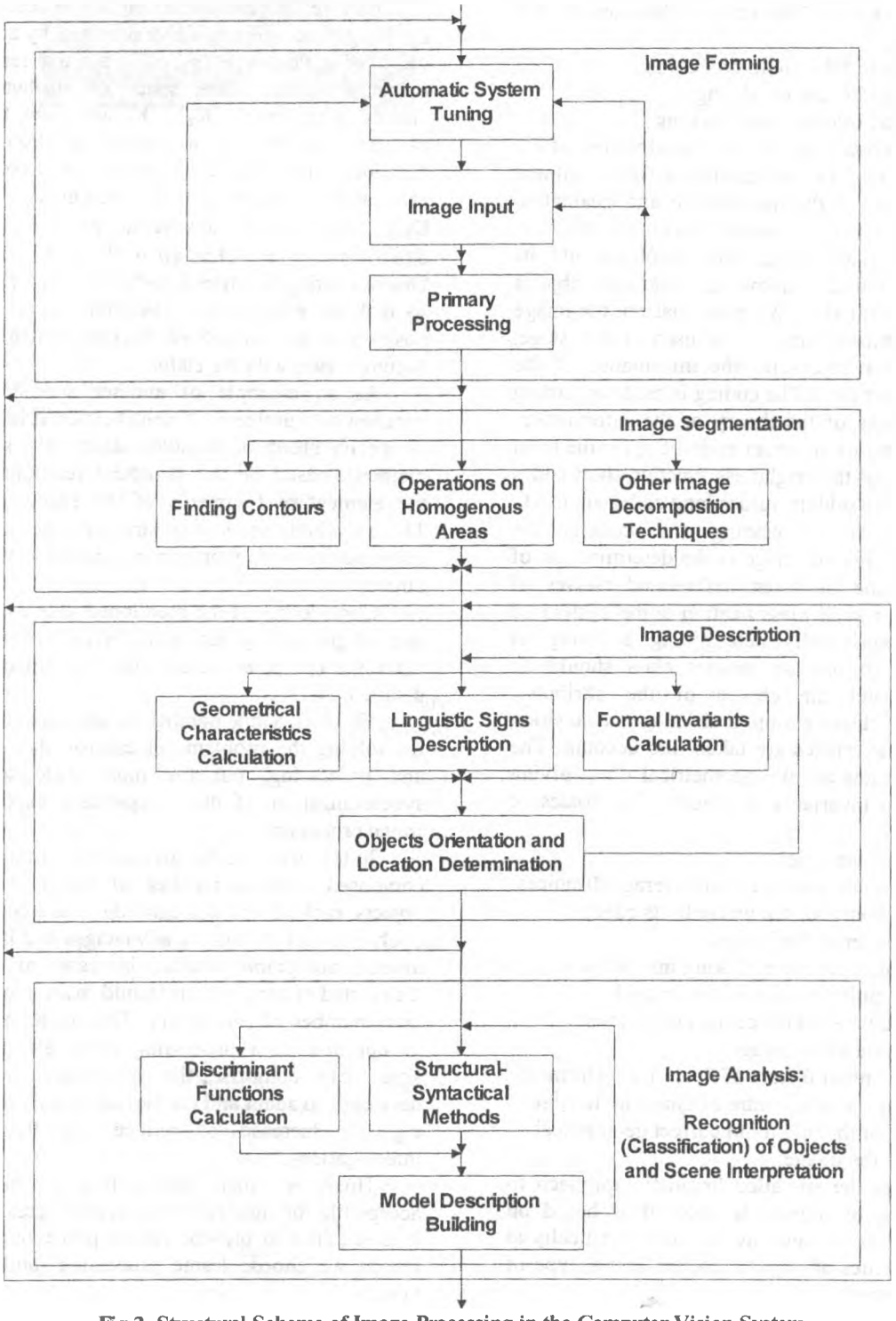


Fig.2. Structural Scheme of Image Processing in the Computer Vision System

in large volume of calculations. The method of spatial differentiation is most frequently applied in practice.

The second approach to the decision of a problem of segmentation is based on the concept of uniformity of the image points, laying in some area, limited by a contour. The basic techniques of this approach are:

1. method of threshold processing;
2. method of area escalating;
3. method of relaxation marking.

The next processing stage is construction of the image description, i.e. representation of brightness function as a set of the quantitative and qualitative characteristics, forming a set of attributes, which is hereinafter used for recognition of object and its classification, and also allows to determine objects location and orientation. We note, that not the image itself, i.e. brightness function, is used at this stage, but its code, that represents the information of the image in a proper kind. The coding is made according to the basic rules of the theory of the information, related to finding of optimum codes. So, for the most frequent values of the brightness function short codes are given, and for seldom values long codes are used.

After primary processing and coding, the analysis of the filtered image is the determination of its fitting to one of given beforehand classes of objects [5]. For such classification some system of attributes, unequivocally determining a fitting of visible object to one or another class should be allocated. During the choice of the attributes, algorithms of classification and image acquisition hardware characteristics are taken into account. The simpliest attributes are the geometrical ones, giving some universal invariants of objects. The basics of them are:

1. area of the objects;
2. maximum, minimum and average distances from centre of inertia of the image to its edge;
3. perimeter of the image;
4. adjusted perimeter (taking into account weight of each point on edge of the image);
5. coordinates of the centres of various moment of inertia of the image;
6. some initial factors of Fourier transform of the distance from inertia centre of image up to edge;
7. sizes of the minimum correct geometrical shape, limiting the image.

Sometimes the so-called linguistic approach to the description of objects is used. It is based on attributes, that are located by human heuristically at study of properties of objects (corner, arrow, type of crossing and other).

The listed geometrical attributes in the majority of cases do not provide the complete description of visible objects, sufficient for their classification, and frequently it is necessary to use more complex formal invariants, for example, moment functions of the image, and other.

As a result of construction of the description of the image each object is characterized by an ordered vector of attributes $p= (p_1, p_2,..., p_n)$, that represents a point in n-dimensional space of attributes. If N classes of objects $K_1, K_2,..., K_N$ are given, the object can be classified by a method of discriminating functions, that in a general case consists in calculation of values of N of functions $d_1(p), d_2(p),... D_N(p)$, such that for any vector $p^* \in K_i$, $d_i(p^*) > d_j(p)$ while $i \neq j$ for all $j = 1, 2, . . . , N$. Discriminating function $d_j(p^*)$ (i=1, 2,..., N) gets out so that the error of classification was as small as possible. A special case of discriminationg functions is comparison with the etalon.

As an example of another approach to the decision of a problem of identification it is necessary to specify group of so-called structural - syntactical methods, based on the structural relations between the elementary fragments of the image of object. These methods use known structures of the discretic mathematics: formal grammars, graphs, networks and other.

Considering of the mentioned above analysis in area of processing and analysis of the images, we make the computer vision system architecture more detailed.

There is a big number of architectural models for solving the problems of control, data collection and processing, but the most wide spread are synchronization of the independent executors and frame processing.

In the first case the architecture of the system is composed from a number of rather independent objects, each of which is carried out as a control flow. Such architecture has the advantages and is, perhaps, unique acceptable model in case of designing distributed system, which should make processing of large number of parameters. This model also allows to optimize data processing more effective (each agent can comprise the information how it is necessary to adapt and change the system parameters, e.g., to increase or reduce the frequency of interrogation).

However, such architecture is not always acceptable for rigid real-time systems creation, when it is required to provide robust processing. For this reason we choose frame processing model for our system.

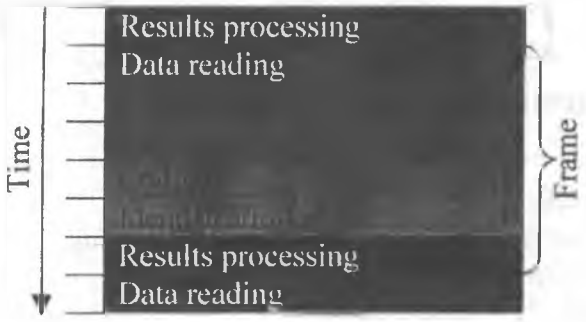


Fig.3. Frame Processing

As it is shown in fig. 3, the process of functioning comes true in this case as a sequence of reading of the image, filtering, segmentation, coding, identification and processing of results through certain intervals of time. Each element of such sequence is a frame. In one's turn, it is possible to divide the frame on a number of sub-frames, taking into account their functional behaviour. The basic advantage of such model is, that we can more rigid supervise a sequence of actions of data collection and processing system.

5. Conclusion

We have proposed architecture of the computer vision software system on the basis of analysis of most common processing methods. This architecture is scalable and flexible in the terms of possibility to use different algorithms of recognition and primary processing of images. Another feature of the offered approach is the multipurpose orientation of the developed architecture that distinct this system from similar ones [6,7].

Our system provides the facility necessary for experimenting with various choices at different abstraction levels to find the best match for the system under development.

6. References

1. G.Booch. Object-Oriented Analysis and Design. Addison-Wesley, New York, 1994.
2. R. Lewis. Practical Digital Image Processing, Ellis Horwood Ltd., Chichester, UK, 1990.
3. U. Seger, H.-G. Graf, and M.E. Landgraf. Vision Assistance in Scenes with Extreme Contrast, *IEEE Micro*, Vol. 13, No. 1, Feb. 1993
4. L.S. Davis. A Survey of Edge Detection Techniques, Computer Graphic and Image Processing, Vol. 4, 1975.
5. R. Duda and P. Hart. Pattern Classification and Scene Analysis, John Wiley & Sons, New York, 1973.
6. J.M. Sanchiz, F. Pla, and J.A. Marchant. A Framework for Feature-Base Motionrecovery in Ground Plane Vehicle Navigation, Proc. Computer Analysis of Images and Patterns '97, Springer-Verlag, New York, 1997
7. VISION Computer Vision System. http://www.llnl.gov/eng/ee/erd/siprg/vision.html.

The research was supported by the INTAS (project № 97-2028).

Walsh Operators of Edge Detection for Gray-Scale Images

Machnev A.G., Selikhanovich A.M.
Institute of Engineering Cybernetics, National Academy of Sciences of Belarus
System Identification Laboratory
Surganova str. 6, 220012, Minsk, Belarus
sel@newman.basnet.minsk.by

Abstract

Walsh operators are proposed that may be used in image processing for edge detection. These operators based on discrete two-dimensional Walsh functions of appropriate dimensionality. The algorithms of a calculation of operators and the algorithm of edge detection are represented.

1: Introduction

Edge detection is an important process in low level image processing. The result of edge detection is initial data for further image processing. During this process pixels that belong to object borders are marked. There are numerous methods; to mention only a few: Prewitt, Kirsh and Sobel gradients, Haralick method, Laplacian of Gaussian, etc. [1-3]. It is difficult to estimate an effectiveness of the methods because of difficulties in determination of the best parameters, bound with each method, and also for lack of a uniform effectiveness criterion. [1, 4]. Therefore the different operators can be effective for various types of images. Below, the Walsh operators are submitted for edge detection on gray-scale images. The effectiveness of these operators is determined by presence of the fast algorithms for their computation. At the same time the edges are determined with the same quality that is obtained using known operators.

2: Walsh operators and their computation algorithms

The two-dimensional Walsh functions can applied for edge detection directly. The four such functions are illustrated in Fig.1 [5]. Thus, the values of coefficients C_{01}, C_{10}, C_{11} corresponding two-dimensional Walsh functions $wal(0,1,x,y)$, $wal(1,0,x,y)$, $wal(1,1,x,y)$ are calculated when direct transform is executed, and the coefficient C_{00} is assumed by zero. When the inverse transform is executed, only the value of coefficient C_{00} is calculated. The coefficients C_{01}, C_{10}, C_{11} can take either on modulo or their true values for computation the value of coefficient C_{00}.

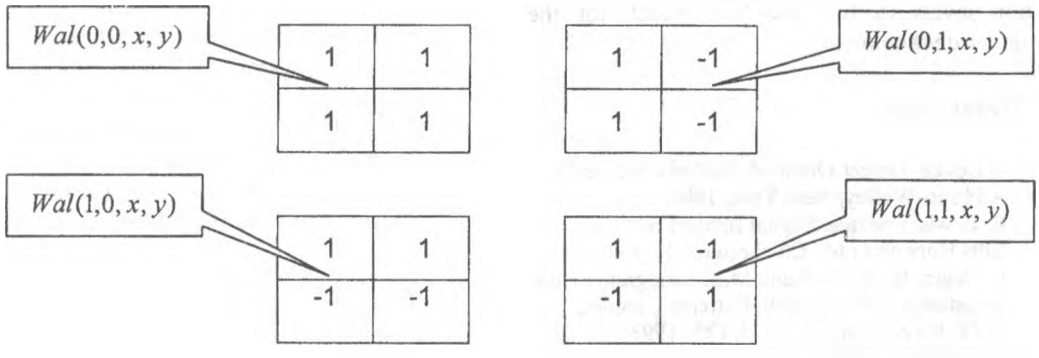


Fig. 1. Two-dimensional discrete Walsh functions.

The differences will consist in a distribution range of intensities of the image obtained after transformation. If the true values are used then the calculated coefficient C_{00} must taken on modulo. At usage of such algorithm the all image area is scanned by 2×2 window with a step equal 1. The value of coefficient C_{00} is assumed to the pixel that corresponds to this coefficient in operator window. A gray-scale image is obtained after applying this algorithm. Edge points are selected by thresholding the magnitude of the coefficient C_{00}.

Two matrix operators by dimensionality 3×3 pixels are usually used for edge detection [1,2]. The Walsh operators of such dimensionality can be obtained from two-dimensional discrete Walsh functions. To this end the line and column with the zero elements are inserted in two-dimensional discrete Walsh functions. The obtained operators are represented in Fig.2.

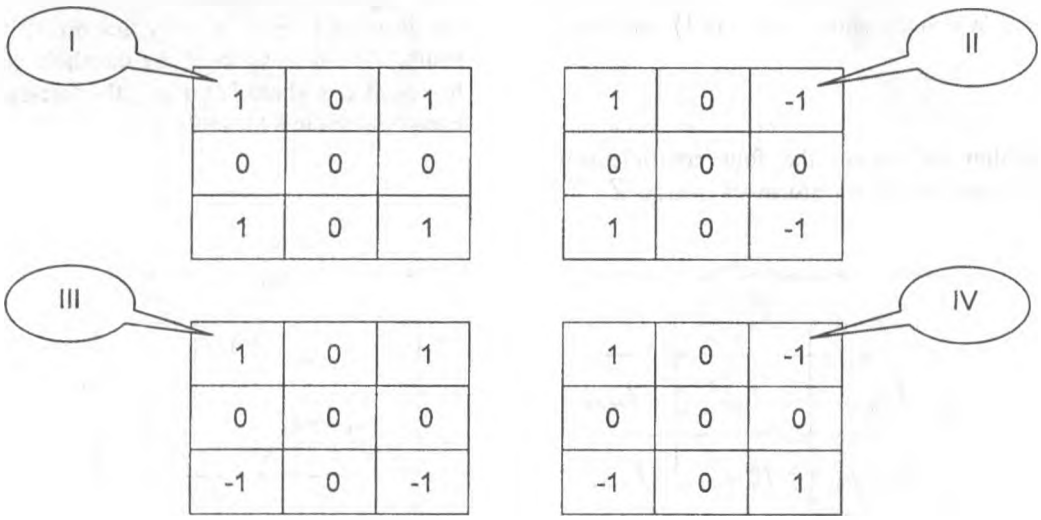


Fig. 2. The Walsh operators.

The Walsh operators are used as well as the known operators. For example, the Sobel operator is applied as follows. The Sobel operator $S(m,n)$ [1,2] involves a nonlinear computation of pixel values over 3×3 window for transformation of the pixel at point (m,n) (see Fig.3). It is defined as either

$$S(m,n) = \sqrt{d^2_{x(m,n)} + d^2_{y(m,n)}} \qquad (1)$$

or for computational efficiency,

$$S(m,n) = \left|d_{x(m,n)}\right| + \left|d_{y(m,n)}\right|, \qquad (1a)$$

where

$$d_{y(m,n)} = (f_{m+1,n-1} + 2f_{m+1,n} + f_{m+1,n+1}) - (f_{m-1,n-1} + 2f_{m-1,n} + f_{m-1,n+1}), \qquad (2)$$

$$d_{x(m,n)} = (f_{m-1,n+1} + 2f_{m,n+1} + f_{m+1,n+1}) - (f_{m-1,n-1} + 2f_{m,n-1} + f_{m+1,n-1}). \qquad (3)$$

The operator is shifted exhaustively over the image area.

Two various combinations of operators represented in Fig. 2 can be used for edge detection with application of Walsh operators. The edges are determined with the best quality when the operators II and III are applied.

The value of the pixel at point (m,n) with application of operators II and III is calculated as follows:

$$W(m,n) = \sqrt{X^2 + Y^2}, \qquad (4)$$

or for computational efficiency,

$$W(m,n) = |X| + |Y|, \qquad (5)$$

where

$$X = (f_{m-1,n-1} + f_{m+1,n-1}) - (f_{m-1,n+1} + f_{m+1,n+1}),$$
$$Y = (f_{m-1,n-1} + f_{m-1,n+1}) - (f_{m+1,n+1} + f_{m+1,n-1}). \quad (6)$$

The fast computation algorithm can be used for a determination of values X и Y corresponding to Walsh operators II and III accordingly. This algorithm is represented in Fig.4 [6] for basis of two-dimensional Walsh functions (see Fig.1) and matrix $Z = \begin{bmatrix} z_1 & z_2 \\ z_3 & z_4 \end{bmatrix}$.

This algorithm determines the four coefficients of two-dimensional Walsh transform of matrix Z. The represented fast algorithm can be used for the operator window with dimension 3×3 pixels (see Fig.3). To this end the input data must be organized by appropriate way. The computation algorithm of Walsh operators II and III is shown in Fig. 5. At usage of these operators the all image area is scanned by 3×3 window with a step equal 1. A gray-scale image is obtained after applying operator. Edge points are selected by thresholding the magnitude of the $W(m,n)$. Fig. 6 shows the edge images obtained by applying Walsh and Sobel operators. The selection of the threshold level is very important to the final results. The discussion of the threshold selection may be found elsewhere [7]. Here, the threshold level has been selected in a heuristic way.

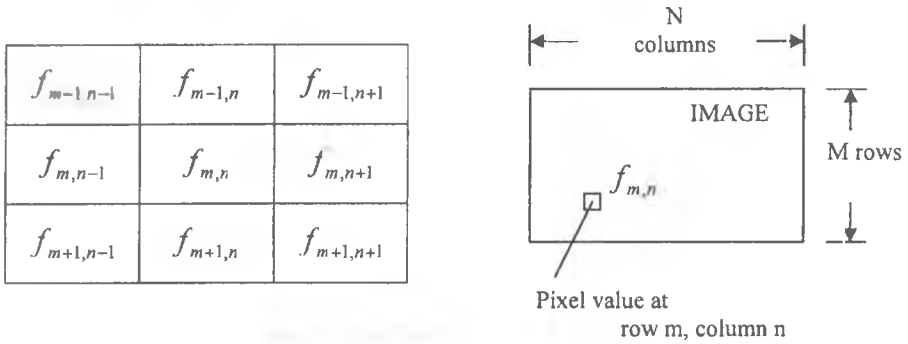


Fig. 3. Notation for an operator of edge detection.

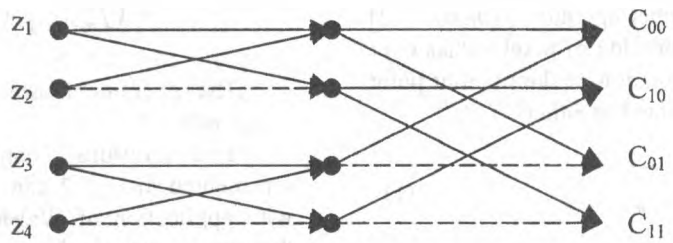


Fig. 4. Fast computation algorithm in basis of two-dimensional Walsh functions.

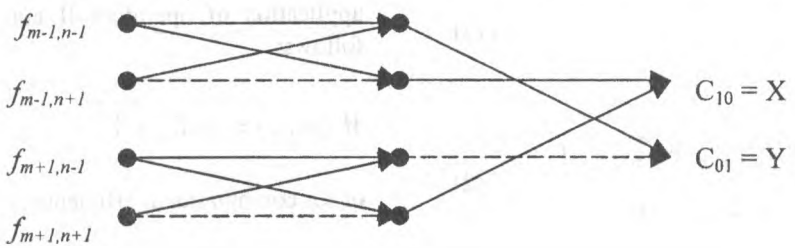
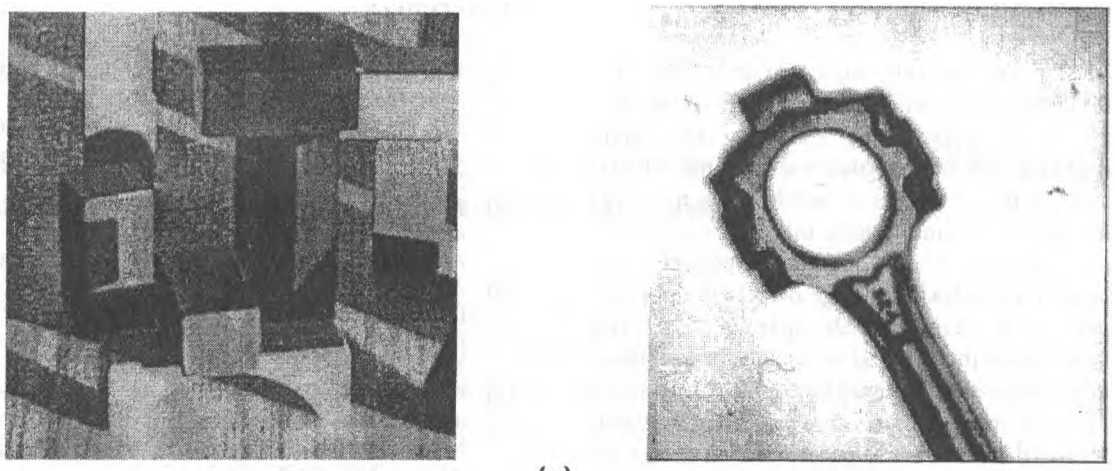
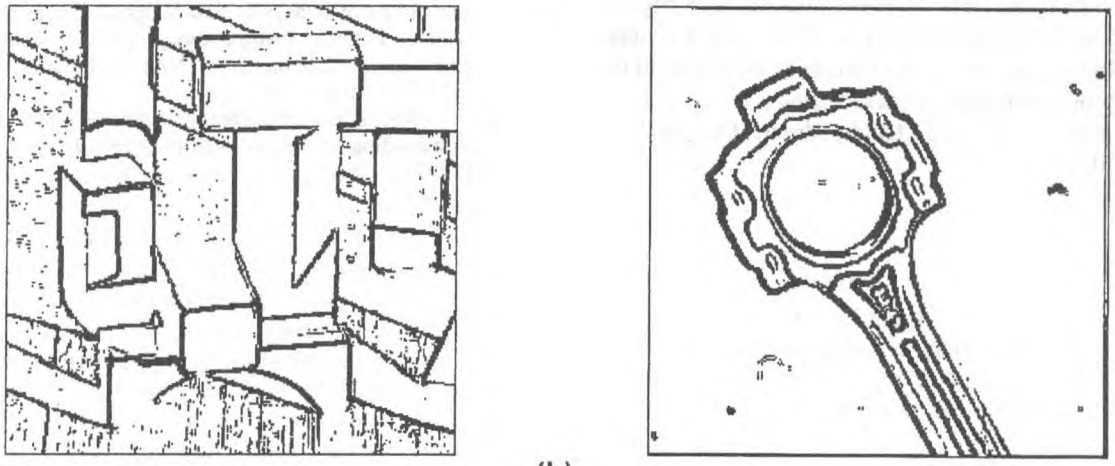


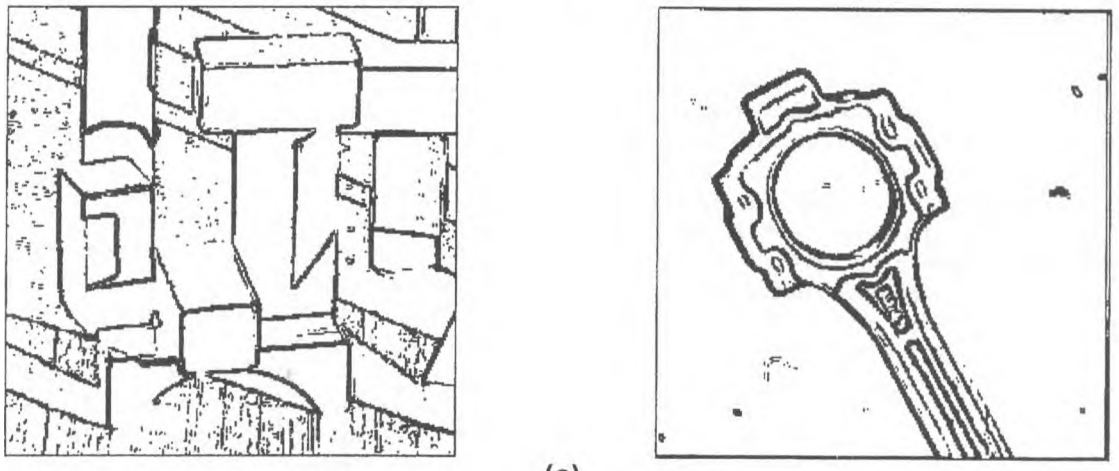
Fig. 5. Fast computation algorithm of Walsh operators.



(a)



(b)



(c)

Fig. 6. Edge images obtained by different operators: (a) original gray-scale images, (b) Sobel operator, (c) Walsh operator.

3: Conclusion

A fast edge detection algorithm based on two-dimensional Walsh function has been developed and reported in this paper. The following two main features distinguish it from known algorithms. Firstly, processing an image by 2×2 window the proposed algorithm allows obtaining more thin contour lines of objects as against Sobel operator. Secondly, the computational complexity of edge detection algorithm is reduced with using Walsh operator. The fast computation algorithm is used advisable for execution direct and inverse Walsh transform. Only six addition operations are required in case of usage Walsh operators. At the same time, processing an image by 3*3 window our operator gives the edge images that identical to those ones by Sobel operator (see Fig. 6). Note, that the algorithm may be effectively hardware implemented and can be used for edge detection in the image processing systems of real time.

The research was supported by the INTAS (project № 97-2028).

References

[1]. W.K. Pratt. Digital Image Processing. John Wiley & Sons, Inc. 1978

[2]. K.S. Fu, R.C. Gonzalez, C.S.G. Lee. Robotics: Control, sensing, vision and intelligence. New York etc.: McGraw-Hill, 1987. – XIII.

[3]. P.C. Yuen, G.C. Feng, J.P. Zhou. A contour detection method: Initialization and contour model.// *Pattern Recognition Letters*, Vol. 20, (1999) pp.141-148.

[4]. A. Koschan, A Comparative Study on Color Edge Detection. // *Reprint from Proceedings 2nd Asian Conference on Computer Vision ACCV'95*, Singapore, 5-8 December 1995, Vol. III, pp. 574-578.

[5]. R.Kh. Sadykhov, A.G. Machnev. Systolic arrays of digital image processing in two-dimensional bases. // Minsk: Institute of Engineering Cybernetics of AS of Belarus, 1996 (In Russian).

[6]. A.G. Machnev, A.M. Selikhanovich. Fast orthogonal transforms of images in two-dimensional bases. // The digest: Pattern Identification, Minsk: Institute of Engineering Cybernetics of NAS of Belarus, 1999. – pp. 37-43 (In Russian).

[7]. P.L. Rosin, Edges: saliency measures and automatic thresholding. // *Mach. Vision Appl.* 9, pp. 139-159 (1997).

Anomalies of Reflection of Acoustic Pulses from Boundary with Strong Dissipative Medium

D. A. Kostiuk, Ju.A.Kuzavko
Brest Polytechnic Institute
224017, Brest, Moscowskaya str., 267, Republic of Belarus
e-mail cm@brpi.belpak.brest.by

Abstract

In given work the longitudinal acoustic waves reflection from boundary of a solid body with dissipative liquid theoretically is considered. The essential dependence of factor of signal reflection and its phase from factor of absorption of ultrasound in dissipative medium is shown. The experimental confirmation of conclusions of the theory is carried out by consideration of reflection of an acoustic pulse from the boundary of plexiglass - epoxy pitch while the last is loaded, shown essential reduction of reflection factor and of acoustic pulse duration while epoxy pitch is hardening.

Keywords: acoustic wave, reflection, dissipative medium.

1. Theory

The reflection of continuous and pulse acoustic signals from boundary of mediums is investigated theoretically and experimentally rather in detail [1]. Nevertheless, the case of reflection of an acoustic wave from medium having strong absorption of sound waves is unknown to us and can appear interesting both in scientific, and in the practical plan. In that work we consider reflection of an acoustic longitudinal wave (LW) from flat boundary of a solid body with strong dissipative medium, as which the viscous liquid can serve.

Let continuous harmonic LW is spread in a solid body without attenuation, which at normal fall on boundary with a viscous liquid partially is reflected, and past LW in a liquid rather quickly fades (fig. 1).

The wave equation for LW in dissipative medium looks like:

$$\rho \ddot{u}_x = c u_{x,xx} + b u_{x,xx,t} \quad , \qquad (1)$$

where u_x - component of longitudinal displacement in LW, c - elasticity module, ρ - density, b - dissipative losses parameter, determined in factors of shift η and volumetric ξ viscosity and thermal conductivity factor χ according to a ratio [2]:

$$b = \tfrac{4}{3}\eta + \xi + \chi(c_v^{-1} + c_p^{-1}) \quad , \qquad (2)$$

in which c_p and c_v are thermal capacities of medium at constant pressure and volume accordingly.

Thus factor of absorption of sound α is unequivocally expressed through the parameter of dissipative losses b according to expression $\alpha=\omega^2/2\rho S_l^3$, where $\omega=2\pi f$ - cyclic frequency of a sound wave, S_l - speed of a longitudinal sound. Let's note, that at $b=0$ these equation determines the acoustic oscillations in a solid body with the appropriate material constants.

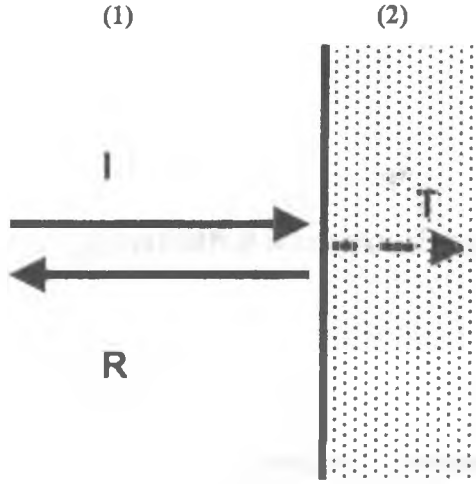


Fig.1. Reflection of an acoustic signal from boundary of mediums

The decisions for falling, reflected and past waves are searched in a standard kind [2]:

$$u^I = u^I_{01}\exp[i(k_1 x - \omega t)]$$
$$u^R = u^R_{01}\exp[i(-k_1 x - \omega t)] \quad (3)$$
$$u^T = u^T_{02}\exp[-\alpha x + i(k_2 x - \omega t)]$$

where $k_1=\omega/S_{l1}$, $k_2=\omega/S_{l2}$ - wave numbers, S_{l1} and S_{l2} - speed of a longitudinal sound in a solid body (1) and liquid (2), t - time.

The boundary conditions at $x=0$ are representing the continuity of displacement and stress in an acoustic wave and will be written down as follows:

$$u^I_x + u^R_x = u^T_x,$$
$$c_1(u^I_{x,x} + u^R_{x,x}) = c_2 u^T_{x,x} + b_2 u^T_{x,xt} \quad (4)$$

That decisions (3) satisfy to the appropriate wave equations, and being substituted in (4), give the system of the linear equations to define the factors of reflection $R = u^R_{01}/u^I_{01}$ and transition $T = u^T_{02}/u^I_{01}$ (T=1+R). Reflection factor has the following kind [3]:

$$R_\omega = \frac{R_0\left[1+(1+x^2)^{1/2}\right]+\frac{T_0}{2}x^2 + i\frac{T_0}{2}x(1+x^2)}{1+(1+x^2)^{1/2}+\frac{T_0}{2}x^2 + i\frac{T_0}{2}x(1+x^2)} \quad (5)$$

where $R_0=(Z_2-Z_1)/(Z_2+Z_1)$ and $T_0=2Z_2/(Z_2+Z_1)$ are reflection and transition factors of acoustic wave accordingly (when $\omega \to 0$), $x=\omega/\omega_c$, $Z_1=\rho S_{l1}$ and $Z_2=\rho S_{l2\,0}$ are acoustic impedances of solid and liquid mediums (without dissipation), $\omega_c=\rho_2 S^2_{l2,0}/b$ is some effective frequency to characterize the dissipative medium, $S_{l2,0}$ is sound velocity (when $\omega=0$).

Starting from (5), a statement for a reflected signal phase can be followed:

$$tg\Psi^R_\omega = \frac{(1-R_0)T_0 x(1+x^2)^{1/2}\left[1+(1+x^2)^{1/2}\right]}{2R_0\left[1+(1+x^2)^{1/2}\right]^2 + T_0(1+R_0)x^2\left[1+(1+x^2)^{1/2}\right]+\frac{T_0^2}{2}x^2(1+2x^2)} \quad (6)$$

Thus, accordingly to (5) and (6) at reflection of an acoustic wave from dissipative medium its amplitude and phase varies.

Below the factor of passage T_ω and phase Ψ_ω of the passed LW are also shown:

$$T_\omega = \frac{(1+R_0)\left[1+(1+x^2)^{1/2}\right]+T_0x^2 + i\frac{T_0}{2}x(1+x^2)^{1/2}}{1+(1+x^2)^{1/2}+\frac{T_0}{2}x^2 + i\frac{T_0}{2}x(1+x^2)^{1/2}} \quad (7)$$

$$tg\Psi_{\omega}^{T} = \frac{\frac{T_0}{2}x(1+x^2)^{1/2}\left[R_0(1+(1+x^2)^{1/2})+\frac{T_0}{2}x^2\right]}{2R_0\left[1+(1+x^2)^{1/2}+\frac{T_0}{2}x^2\right]+\left[(1+R_0)(1+(1+x^2)^{1/2})+T_0x^2\right]+\frac{T_0^2}{4}x^2(1+x^2)} \quad (8)$$

2. Computer experiment

Proceeding from the given dependence R_ω and using direct and inverse Furier transformations with the help of the computer the form of the reflected signal from boundary of plexiglas - epoxy pitch was estimated for model and real acoustic pulses. The results are given in a fig. 2, 3. The changes of amplitude and phase can be unequivocally connected to properties of a contact liquid and superficial layers of a body and contra-body, for example in tribological research of pairs friction.

If the reflection occurs from less dense acoustic medium ($Z_2 < Z_1$), at $\omega << \omega_c$ then there is an inversion of a signal ($\Psi^R = \pi$). In a vicinity $\omega \sim \omega_c$ the minimum of reflection factor is observed at the further increase of a phase of the reflected signal concerning a phase of a signal, falling on boundary. Further at $\omega >> \omega_c$ $R_\omega \to 1$ and $\Psi^R \to 2\pi$. There is a complete reflection of a signal. Otherwise at reflection from more dense medium the inversion of a signal does not occur ($\omega >> \omega_c$, $R_\omega \to R_0$ and $\Psi^R \to 0$). Similarly at $\omega \sim \omega_c$ the minimum of reflection factor R_ω is observed at a maximum of a phase Ψ^R_{ω}. Further at $\omega >> \omega_c$ $R_\omega \to 1$ and $\Psi^R \to 0$.

3. Real experiment

To confirm the theoretically predicted above phenomenon - dependence of the reflection factor from the dissipation of ultrasonic energy in reflecting medium the following experiment was carried out. The pulse generator feeds ultrasonic piezoceramical transducer (UPT) with resonance frequency of 3.5 MHz. An acoustic pulse close to the theoretically considered form was radiated into the structure of plexiglass - epoxy pitch. Radiated and reflected signals were registered by oscillograph.

In a fig. 4 the dependence of reflection factor R of a pulse signal in arbitrary units is presented during hardening of epoxy pitch prepared accordingly to the state standard (10 g of epoxy pitch to 1.2 g of curing agent). Let's note that acoustic impedances of liquid and hard phase of epoxy pitch are differing no more than 100%. During the mix hardening temperature grew no more 10^{O}C in comparison with room, that practically did not influence on the acoustic parameters of the mix. It is possible to explain the reduction of reflection factor in 2,5 times on our sight only by the theory advanced here, namely sharp change of energy dissipation in an epoxy layer while hardening. In a fig. 3 theoretical dependence of reflection factor (represented by dot lines) at $Z_1 = 3.1 \cdot 10^6$ kg/(m^2 s), $Z_2 = 3.25 \cdot 10^6$ kg/(m^2 s), for the basic frequency of a pulse signal at ω_c=10 MHz for a solid phase. Also at hardening of the epoxy pitch the duration of the reflected acoustic signal changed from τ=3 μs up to τ=2 μs, that will be coordinated to conclusions of the advanced here theory.

The most unexpected results appeared when on 4 volumetric parts of epoxy pitch 1 volumetric part of curing agent was added. The process of hardening of epoxy pitch occurred very intensively and non-uniformly with the change of volume up to 10 % and increase of temperature of a mix up to 80^{O} C. The results of experimental researches are given in a fig. 5, whence it be visible, that the change of reflection factor of an acoustic pulse signal had maximum change in 8.5 times and qualitatively coincided with the advanced here theory. For the quantitative coordination of the theory with experiment the study of dynamics of viscosity and of some physical-chemical factors of hardened epoxy pitch is necessary, that is an independent task and serious experimental efforts are required for that.

It is necessary to make a conclusion, that the condition of reflecting dissipative medium essentially influences the reflection factor and phase both continuous and pulse acoustic signals. As the phase measurements are rather exact in comparison with amplitude ones, so on them we can estimate the sound absorption in strong dissipative medium and to carry out the direct measurements of viscosity of liquids.

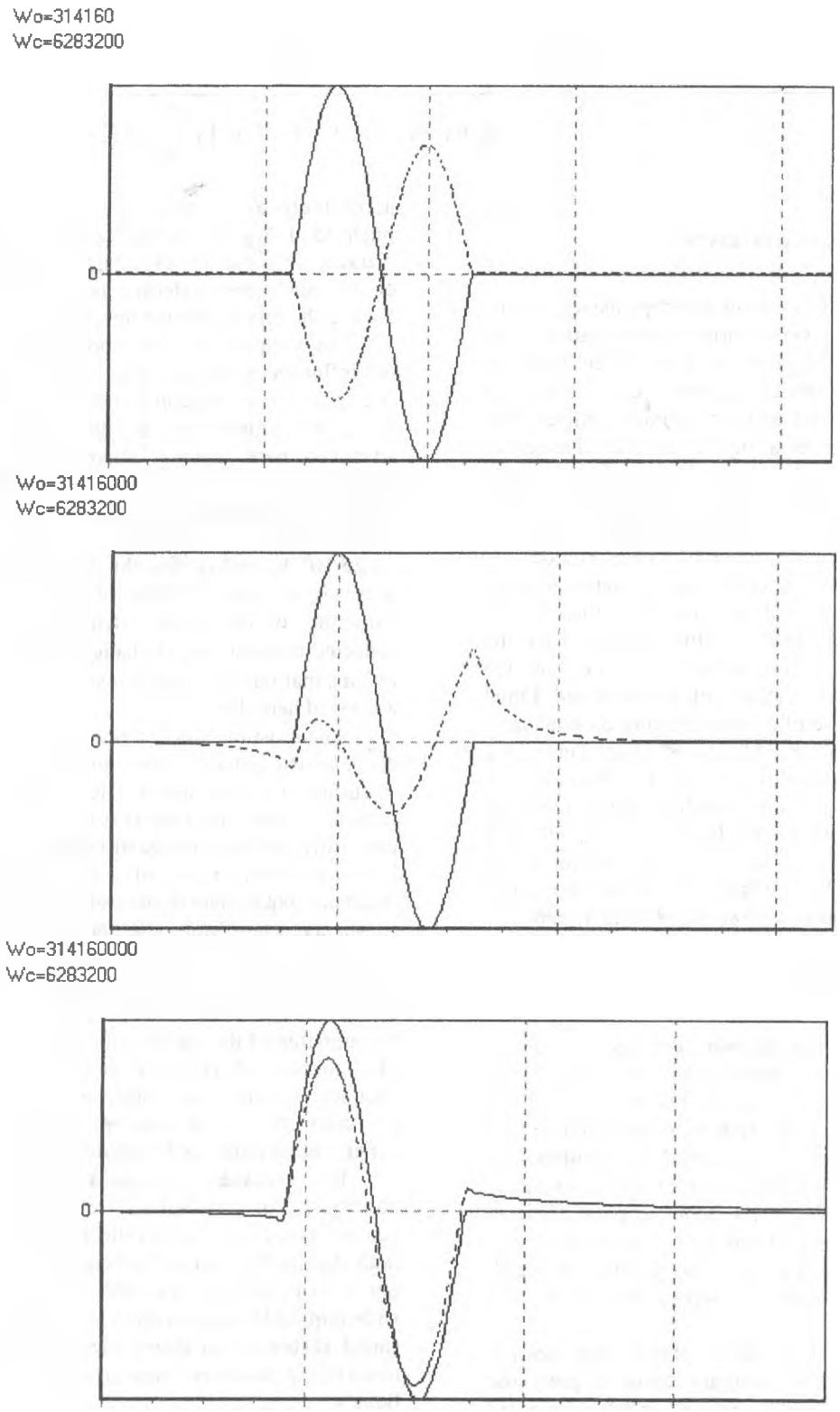
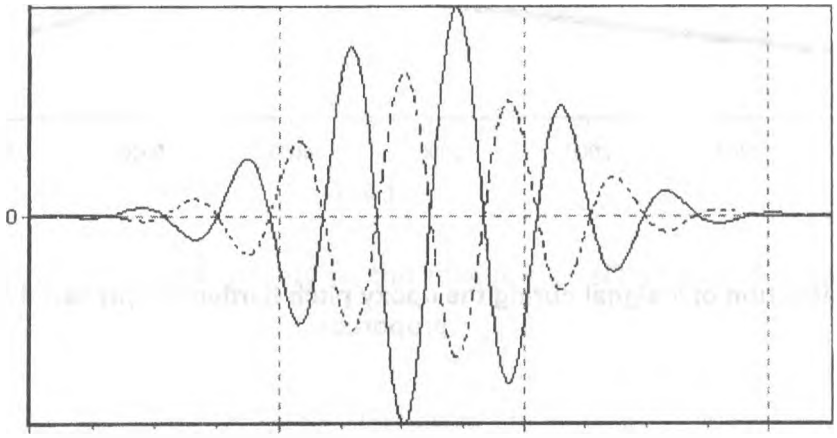
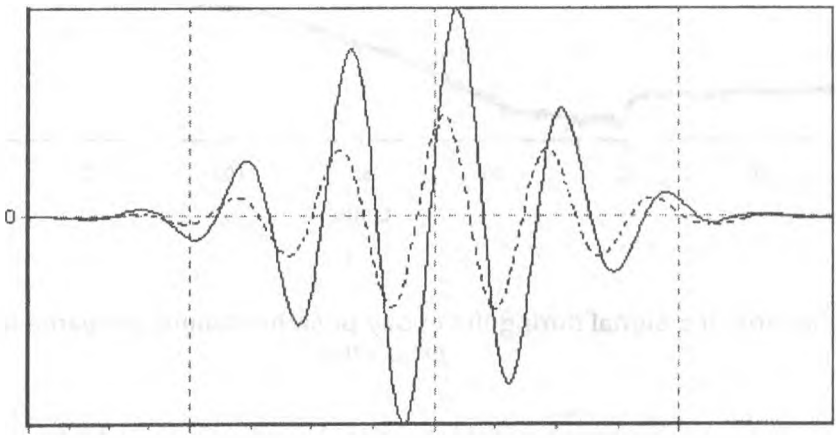


Fig. 2. Reflection of model pulses:
—— falling pulse, ····· reflected pulse

Wo=314160
Wc=6283200



Wo=31416000
Wc=6283200



Wo=314160000
Wc=6283200

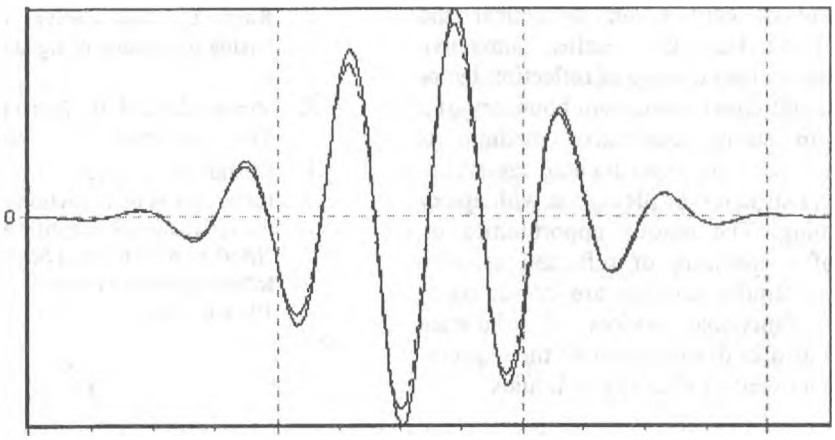


Fig. 3. Reflection of real pulses:
____ falling pulse, ... reflected pulse

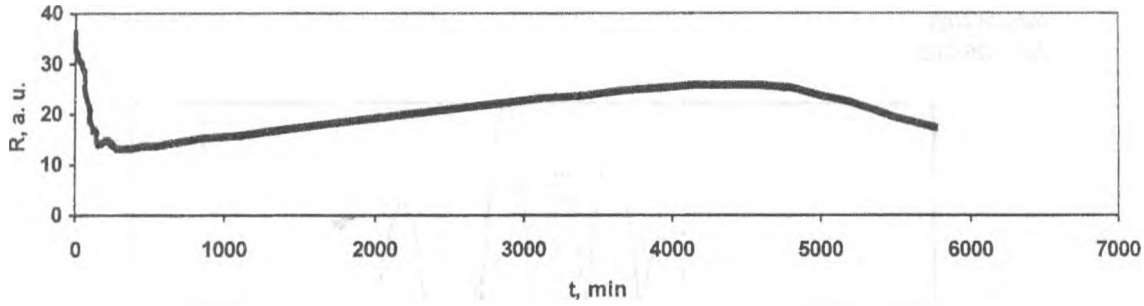


Fig. 4. Reflection of a signal during the epoxy pitch hardening, prepared in 10/1.2 weight proportion.

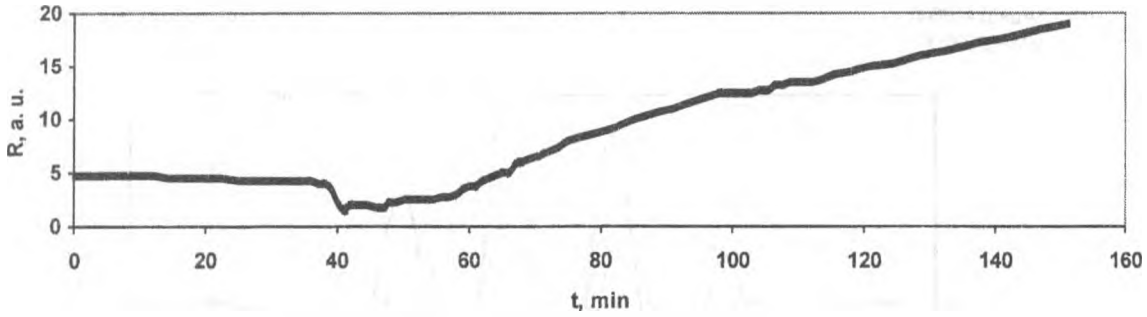


Fig. 5. Reflection of a signal during the epoxy pitch hardening, prepared in 4/1 volumetric proportion.

Conclusion

As a result of carried out theoretical and experimental researches the earlier unknown phenomenon - anomalous change of reflection factor of an acoustic longitudinal wave from boundary of a solid body with strong dissipative medium is established. Its model in experimental researches was the two-layer structure of plexiglass with epoxy pitch at hardening. The unique opportunities on measurement of a spectrum of reflected acoustic signals in such or similar structure are interesting in development of functional devices of solid-state electronics, and also in development of the express-method of measurement of viscosity of liquids.

References

1. Kayno S. Acoustic waves. Devices, visualization and analog processing of signals / M.: "Mir". 1990. 656 p.
2. Vinogradova M. B., Rudenko O. V., Suhorukov A. P. Theory of waves / M.: "Science". 1990. 432 p.
3. Kostiuk D. A., Kuzavko Ju.A. Acoustic pulse researches at pairs friction modeling. / Materials of the International scientific and technical conference "Modern directions of development of industrial technologies and robotics ". Mogilyov. Belarus. 1999. p. 186.

Program improvement of quality of the images at 2D acoustic visualization

Ya. A. Akulich, D. A. Kostiuk, N. V. Kudinov, Yu. A. Kuzavko
Brest Polytechnic Institute
224017, Brest, Moskovskaya str., 267, Republic of Belarus
e-mail: cm@brpi.belpak.brest.by

Abstract

The various program methods are offered (linear filtration, processing on special functions with an adjustable weights, color-coding) to improve the images quality at 2D acoustic visualization of solid-state products, which are illustrated on an example of researches of friction pair. Overlapping the acoustic images (AI) with the appropriate optical images (OI) with required scale transformation in addition is realized. The quantitative criteria of an estimation of the characteristics of a contact stain of friction pair are offered.

Keywords: ultrasonic diagnostic complex, acoustic and optical image, filtration, scale transformation, pair of friction.

1. Introduction

The development of computer facilities has resulted in significant improvement of systems of ultrasonic diagnostics. Now at realization of non-destructive testing frequently there is a necessity for reception of the acoustic image with the high resolution. To receive the necessary information at scanning object with use of acoustic waves and to generate its visual image it is obviously possible through systems of ultrasonic visualization.

The methods of ultrasonic visualization already have received wide application in various areas of a science and engineering: non-destructive production quality surveillance, medical diagnostics, diagnostics of materials and products. However, in researches directed on maintenance of reliable functioning of mechanisms, technical diagnostics both control of processes of friction and wear process are important, the application of methods of ultrasonic visualization seems to be new and rather perspective.

2. Estimation of the tribological characteristics

The ultrasonic diagnostic tomography is carried out through the ultrasonic diagnostic complex (UDC), which consists of ehotomoscop ETS-U-02 with ultrasonic piezoceramical transducers (UPT), a videoadapter, a personal computer (PC) with a laser printer and an optical scanner. The mechanical cross section scanning of an object under control placed in a working fluid is carried out with the help of the UPT. The echo-signals received undergo preliminary processing and are displayed on the screen of the monitor echotomoscop as a cross section AI of the object.

From the output of the echotomoscop AI is sent through the videoadapter to the PC. The optical image of the drawing of the examined section of the object is entered in to PC with the help of the optical scanner. Then images are overlapped and processed with the help of a special program.

As a result of comparison of the processed images it is possible precisely to look after interrelation of dynamics of change of AI parameters with the change of friction pair loading.

When friction pair is on loading and rotation there was a luminous contact stain on AI, representing 32 gradation of gray color. Area and intensity of that stain were counted accordingly to listed below algorithm, that allowed to compare changes of received AI to changes of parameters of researched object.

In a product representing the "shaft-and-bush" type friction pair, as it is shown in a fig. 2, a surface of their contact can be effectively diagnosed. If in this area there are cracks or local-intense condition, there is on two - three order a large area in which deformation have changed density and elasticity module of substance. So conditions of reflection of a sound are changing that entail also AI changes. As last is a matrix of 512x512 points with 32 gradation of brightness, it is possible to allocate on it the rectangular containing a contact stain. Brightness for all allocated area according to a ratio further is encountered:

$$I = \sum_{i=1}^{N} \sum_{j=1}^{M} I(X_i, Y_j) \quad , \qquad (1)$$

where M, N - length of the sides of the allocated rectangular, brightness I (X_i,Y_i) - code of brightness of the image point. The size I is represented in arbitrary units of brightness of all rectangular.

If to study dependence of brightness I from the external factors (friction pair loading p, rotation frequency w, etc.) then the criteria of an estimation of tribological properties of such friction pair would arise. Thus the relative dependence seems to be interesting:

$$\Delta I(p, w) = I(p, w) - I(p = 0, w = 0) \quad , \qquad (2)$$

which unequivocally defines the influence of the external factors.

Two remarks should be made. The first chosen rectangular should be inside the second rectangular, if I_{21}(p, w) for addition of crossing of sets of these points is constant, the first rectangular can be narrowed. Thus the process proceeds so long as the condition will not be broken. The size of area subject to operational changes thus is determined. Else process of expansion of the first rectangular should be carried out, until I_{21} will not become a constant. Rectangular can be replaced by any other closed curve. There is some disorder in brightness of points because of echotomoscope features, therefore it is recommended for each point to choose a code I(X_i,Y_i), average for some interval of time.

2. Computer processing of the images

The ultrasonic visualization of friction processes have some difficulties caused by occurrence of false images as a result of reflections and transformations of ultrasonic oscillations in controllable solid-state friction object, for which elimination, the additional digital processing AI is offered.

The various kinds of a linear filtration, processing on special functions with predefined and adjusted weights and color-coding of the black-and-white images [2] were applied to improve the AI quality. Besides the program overlapping AI with its optical analogue was made.

The application of filtration methods was oriented on such tasks as exception of extraneous noise, expression of borders of image elements and correction of its brightness characteristics.

At the decision of a task of noise exception the advantages of a low-frequency filtration overwhelming spatially uncorrelated noise (which contains some higher spatial frequencies in its spectrum comparison with a spectrum of the basic image) were used. In particular, for this purpose the operation of discrete convolution of the initial image with various noise-suppressing masks was used.

Noise smoothing has as a by-effect some blurring of image contours (compare the initial image on fig.1a and result of filtering on fig. 1b), so it was applied in a combination to various techniques of borders expression. Among other the discrete filtration with the high-frequency pulse response (fig. 1c) was used for that.

The same discrete convolution with masks chosen proceeding from a condition of equality to 1 of the sum of their elements was applied to realization of a high-frequency filtration.

For the greater visual expressiveness the emphasis of borders was accompanied by imposing of the optical image received by scale transformation of the scanned drawing of the examined detail.

In addition to the image filtration the correction of its brightness characteristics expressed in mapping of its intensity meanings on a range, definitely distinguished from initial was made. Linear transformation of intensity that is change of general brightness of the image, was made and also gamma

correction, resulting in nonlinear changes of intensity of image elements. Last allowed to correct separate areas of intensity range, achieving more precise emphasizing of required image elements. The final variant of the image is shown in a fig. 2.

Using of color-coded images is associated with feature of human vision, allowing distinguishing the greater number of various color hues, than number of hues of gray color. Hence the transition from black-and-white to pseudo-colors improves the recognizing of low-contrast image elements, and also can be used for visual emphasizing of its separate regions.

Color-coding of the initial images was made in connection to area of a contact stain, the brightness gradation of which received most "hot" color hues (appropriate to gradation of red color), while "heat" of other part of the image decreased with distance.

Use of the optical image imposed on acoustic one allows to discharge the interfering factors, which are the consequence of cross waves occurrence in a solid product, because of what there are false images on AI. The analysis of brightness of the false images on an above mentioned technique can be useful for quantitative criteria of an estimation of physical and tribological characteristics of friction pairs as the transverse waves carry additional information about the contact stain, which cannot be achieved by longitudinal waves.

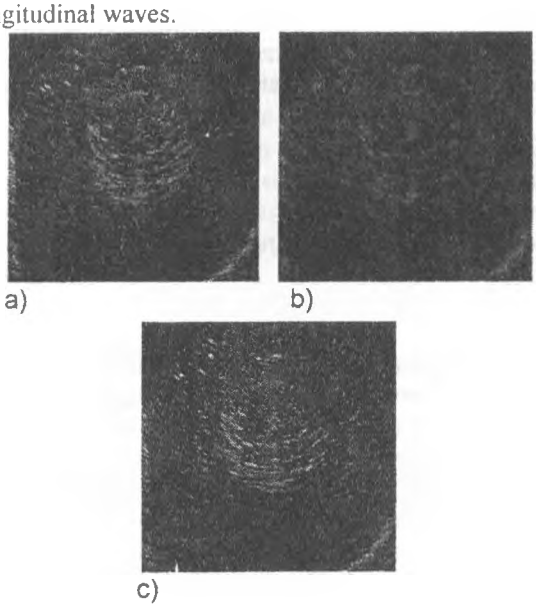


Fig. 1. Application of a linear filtration a) - initial image; b) - result of application of the low-frequency filter; c) - result of application of a high-frequency filter

For the coordination of AI and OI scales it is necessary to reduce the overlapping image size n times ($n=S_m/S_l$, where S_m and S_l are velocities in a material and in a contact liquid accordingly) or to increase the image size in the same proportion. For example, for an aluminum product n = 4.6.

At overlapping AI and OI it is necessary to find two identical points in each of the images. It is obvious, that for AI those are two nearest to emitter. In the worse case there is one, and second is possible to believe the emitter itself. Correctly executed scale transformation makes identical all points of mediums boundary.

Overlapped images are shown on fig. 3.

Fig. 2. Image passed through complex processing.

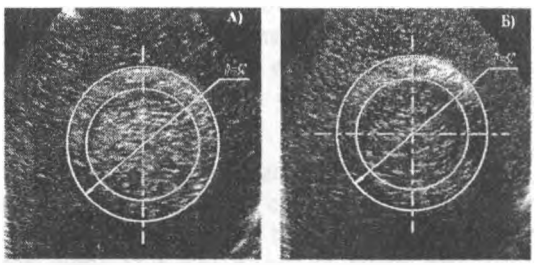


Fig. 3. Images with AI and OI overlapping (AI obtained with different modes of loading) a) Frequency of rotation n=400 min^{-1}; loading P=500 N; b) Frequency of rotation n=400 min^{-1}; loading P=2000 N.

The acoustic images were received with the help of the machine for test of materials for friction and wear 2070 SMT-1. The material of the shaft was served by steel 40H. The bush was made of bronze OCS-5-5-5. This materials have found wide application in condensing devices, bearings of sliding and other units of friction. The industrial oil I-40A was used as a working liquid. The ultrasonic

transducer fastened with special rig at investigated cell of friction machine.

Conclusion

In the whole set of mentioned above transformations has allowed to achieve much higher quality AI, making it essentially more recognizable. Using of (1) and (2) makes it possible to get a quantitative criterion of solid-state products dynamics examination.

References

1. Akulich Ya. A., Gladyshchuk V. B., Golub M. V., Kuzavko Yu. A., Sklipus B. N. Ultrasonic Diagnostic Complex Tomography of Machine-Building Products. // The American-East Europe conference " New materials and technologies in tribology ". Minsk-Grodno-Warsaw. 1997. P. 86
2. Pratt W. K. Digital image processing. v.2. / M.: "Mir". 1982. 790 p.
3. Akulich Ya. A., Kostiuk D. A., Kuzavko Ju.A. Program processing of the acoustic images of friction pairs. // Materials of the International scientific and technical conference "Modern directions of development of industrial technologies and robotics". Mogilyov. Belarus. 1999. p. 186.

The Universal Contact Device for Monitoring Printed Circuit Cards.

Vasiliy Shoot' [1], Igor Prozherin [2]

(1) Department of the electronically-computer and web, The Brest politechnical institute, Belarus, Ph. +375-162-421081

(2) Electronically-mechanical faculty, The Brest politechnical institute, Belarus, Ph. +375-162-416714, E-mail: Igor_Prozherin@p245.f9.n454.z2.fidonet.org

Analysing the reasons hindering creation of the universal contact device, the construction of the universal contact device which has contact pins in all knots of an abacus with a pitch 1.25-mm is developed. The accounts of basic elements and knots of the universal contact device are given.

Recently at the enterprises the large distribution is received with automated control and measuring systems on the basis of managing electronic - computers for monitoring multilayer printed circuit cards. The development and manufacturing of contact devices takes a significant place in all amounts of works on designing and manufacturing of means of monitoring. The attempt of creation of constructions of the universal contact device, which has contact pins in all knots of an abacus with a pitch $A = 1.25\,\text{mm}$, meets large difficulties.

A successful solution of this problem needs serious development of constructions of the most contact element in the plan of its miniaturisation and problems of laying of large number of conductors going to these elements, in view of providing of access to them for repair and replacement of contacts.

By the most simple and reliable solution of the first part of a problem was the fulfilment of a contact element as an elastic rod covered with an elastic dielectric, to which one end face the conductor of the control device is soldered, and on the friend the taper forming dot contact to an inspected surface is executed. This optimal solution was that in one element the functions of collector and of spring being two integral parts of any contact device are combined.

However for want of pitch $A = 1.25\,\text{mm}$ will produce the universal contact device of such construction rather complicatedly because of difficulties of mounting stipulated by a high denseness of a disposition of contact elements on unit of square. So, for want of pitch $A = 1.25\,\text{mm}$ the specific denseness makes $\rho = 64$ *unit on sq.cm*. For this reason it is difficult to check and practically it is impossible to correct a breakaway of a conductor of connection of contact elements with the control device. The probability of a breakaway is rather high, as the contact elements are mobile, and the number them is great.

It is rather complicated to produce the universal contact device of such construction with a pitch $A = 1.25\,\text{mm}$ because of difficulties of mounting stipulated by a high denseness of a disposition of contact elements on unit of square. So, for want of pitch $A = 1.25\,\text{mm}$ the specific denseness makes $\rho = 64$ *unit on sq.cm*. For this reason it is difficult to check and practically it is impossible to correct a breakaway of a conductor of connection of contact elements with the control device. The probability of a breakaway is rather high, as the contact elements are mobile, and the number them is great.

The universal contact device for monitoring multilayer printed circuit cards with a size of a field $B \times L$, in which the necessity of a soldering of conductors to contact elements is eliminated, is represented on Figure 1. It represents the basis 1 with the base pins, fixed on it, (2), on which the package from auxiliary printed circuit cards (3) is installed. The upper, intermediate and lower auxiliary payment are represented on Figure 2-4.

The printed conductor (4) auxiliary payment (3) is supplied with a bonding contact pad (6), on which by the lower extremity the contact element (10) rests, and upper extremity it contacts to the checked payment (12).

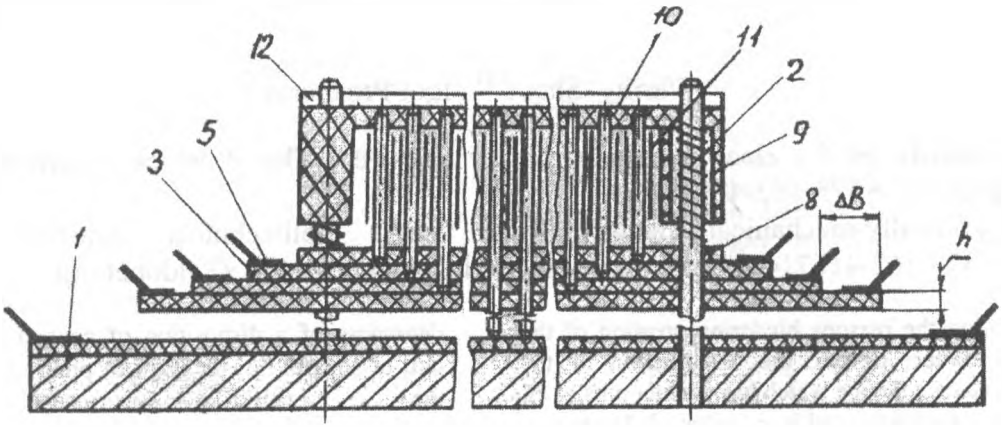


Figure 1. The universal contact device for monitoring printed circuit cards

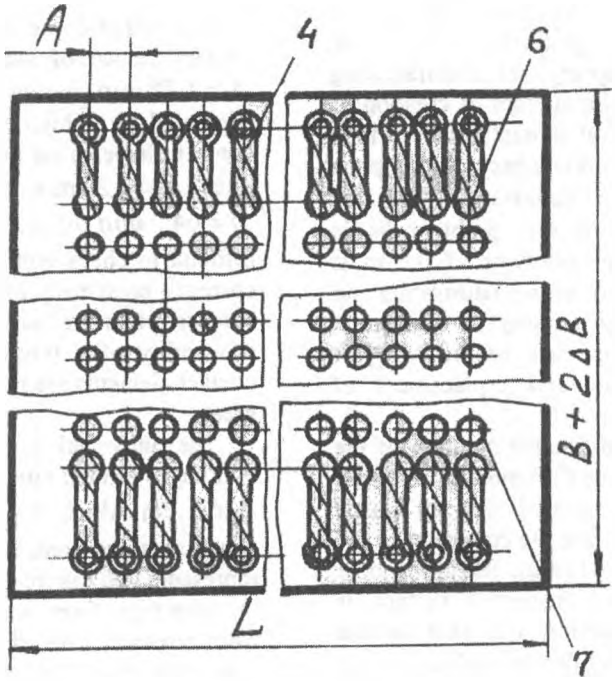


Figure 2. The upper auxiliary payment

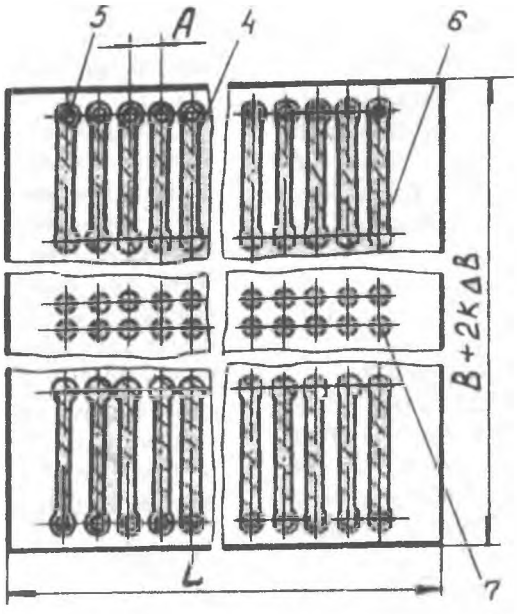


Figure 3. The intermediate auxiliary payment

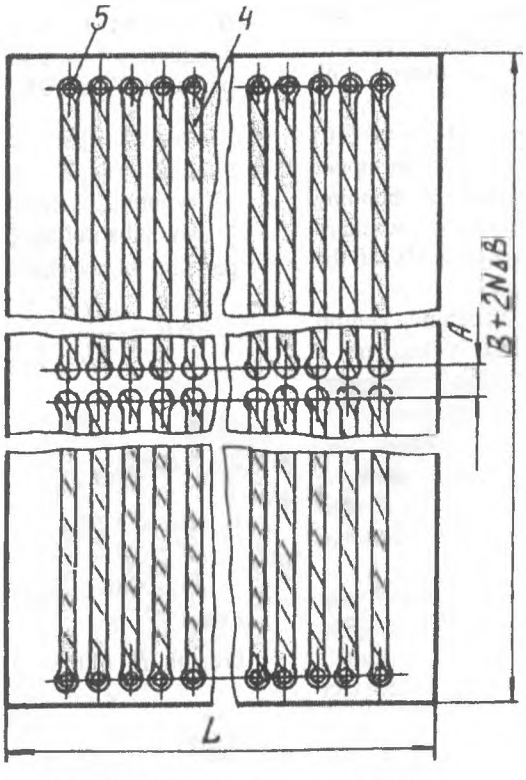


Figure 4. The lower auxiliary payment

On the lower auxiliary payment Figure 4 the distance between pins of bonding contact pads (6) counter printed conductors (4) is equal to a pitch A, and on each consequent payment this distance is more, than on previous on a double pitch $2A$ Abacus. This implies, that the number N Auxiliary payment (3) in a package is determined by expression

$$N = \frac{B}{2A} \qquad (1)$$

Here B - breadth of the inspected payment.

The distance between pins of printed conductors (4) also is equal to a pitch A. On a free field of all auxiliary payment, except for lower, the orifices (7) are executed, which disposition corresponds to a disposition of an abacus of the checked payment (Figure 2, 3).

On base pins (2) the flat (8) and π-figurative (9) guides are installed. Last is installed above first. Guides (8) and (9) have coaxial orifices, which centres correspond to knots of an abacus of the checked payment. In orifices of guides (8), (9) and auxiliary payment (3) packages contact elements (10) are installed, which surface is covered with elastic dielectric (11).

Both central numbers of contact elements (10) pass through orifices in guides (8), (9) in upper auxiliary payment and concern face of bonding contact pads (6) by the one end of lower auxiliary payment, and other - bonding contact pads of the checked payment.

The length of the following behind central numbers of contact elements (10) is less on a thickness of the auxiliary payment. They concern by the lower end face of bonding contact pads (6) on following for the lower auxiliary payment etc. contact elements (10), located in extreme numbers, pass through orifices in guides (8), (9) and lower end face (6) upper auxiliary payment concern bonding contact pads.

The universal contact device for monitoring printed circuit cards works as follows. On base pins (2) of the inspected payment (12) should be installed and pressed to the upper end faces of contact elements (10), which are curved elastically. The dielectric cover (11) protects contact elements (10) from closure among them. Since the lower end faces of contact elements rest on appropriate bonding contact pads (6) printed conductors (4), connected with output pins (5), the bonding contact pads of the inspected payment appear on-line to the electronic device through electrical circuits consisting of contact elements (10), printed conductors (4) auxiliary printed circuit cards of (3) and output pins (5).

Let's reduce necessary accounts of configuration items of the universal contact device, which are useful to its manufacturing.

So each consequent auxiliary payment (3) of the package is wider than previous on magnitude $2\Delta B$, which is required for laying conductors of connection of contact elements with the control device. Let's calculate an increment of a breadth ΔB on one edge of the payment. Height of stacking of conductors is equal to a thickness of the payment h (Figure 1); i.e. the conductors are stacked up to a level of the following payment. Let's make the equation for definition ΔB

$$\Delta B h = \frac{n\pi d^2}{4} \qquad (2)$$

Where d - diameter of used conductors.

The amount of conductors n, which can be put on one ledge on an edge of the control device along it of length L, is determined by number of printed conductors 4 (or bonding contact pads 5) of auxiliary payment (Figure 2).

$$n = \frac{L}{A} - 1 \qquad (3)$$

Where L - length of the inspected payment.

By substituting (3) in (2) and by dividing both parts on h, we shall receive:

$$\Delta B = \frac{\pi d^2}{4h}\left(\frac{L}{A} - 1\right) \qquad (4)$$

The breadth of the contact payment from a package, where $K \in 1..N$ Settles up by following expression:

$$B + 2K\Delta B = B + \frac{K\pi d^2}{2h}\left(\frac{L}{A} - 1\right) \qquad (5)$$

By substituting instead of K In expression (5) value N from (1), we shall receive a dimensional breadth B_1 of the control device:

$$B_1 = B\left(1 + \frac{\pi d^2}{4Ah}\left(\frac{L}{A} - 1\right)\right) \qquad (6)$$

Height H package of the auxiliary payment (3) is determined by following expression:

$$H = Nh = \frac{Bh}{2A} \qquad (7)$$

Conclusions

1. The construction of the circumscribed universal contact device with a pitch of 1.25 mm has high reliability, adaptability to manufacture of manufacturing and maintainability. There are no mobile soldered junctions in it and the denseness of the soldering in some times is lower than in control devices, now in use in production.
2. Obtained in the work the settlement expressions allow to calculate at a stage of designing all necessary sizes of elements of the contact device, with specific basic data B, L, A, defining parameters of the inspected payment.
3. The given device was used in creation of a system of recognition of electronic blocks with the erased marks of integrated chips and unknown topology of multilayer printed circuit cards.

Synthesis of a Test Generator for a Built-In Self-Test Scheme

Oleg V. Khomich ,Vyacheslav N. Yarmolik,
Department of Computer Science,
Belorussian State University of Informatic and Radioelectronic,
6, Brovki Str., Minsk, 220027, Belarus
helgi@belcaf.minsk.by, yarmolik@yvn_poit.minsk.by

Abstract

This paper presents a new algorithm for the automated synthesis of pseudo-random test patterns generators for Built-In Self Test schemes with a mixed test mode. The experimental results show an opportunity of using the given method on a design stage of circuits producing. In this paper it is shown that an appropriate selection of test pattern generator can significantly reduce the hardware requirements of deterministic part.

1. Introduction

Using the Built-In Self Test schemes is one of major methods for reliable functioning of circuits. The efficiency of Built-In Self Test is estimated by duration of the test and hardware overhead for a test scheme realization.

Nowadays most approaches to diagnosing circuits based on exhaustive testing, pseudorandom testing, weighed random testing, hardware patterns generators of deterministic tests and mixed-mode test pattern generation frequently are used [1-4]. Practically in all named methods wide application finds the Linear Feedback Shift Register (LFSR) as the generator of pseudo-random test patterns. LFSR has simple structure which requier small area overhead, and its can also be used as output response analyzer thereby serving a dual purpose.

Unfortunately, the majority of the combinational circuits contain random pattern resistant faults. In such cases for achievement a maximal fault coverage both pseudo-random and deterministic patterns are used. [5-7].

In paper[3] a scheme for Built-In Test with multiple-polynomial LFSR is offered. The presence of the mechanism for choice necessary polynomial for realization of the generator of pseudo-random patterns allows considerably improve fault overhead in comparison with use traditional LFSR as the generator. However, the management of a choice of a particular polynomial requires in this case additional hardware expenses.

In article [8] the method of updating generated pseudo-random test patterns is offered with the purpose of increase a fault coverage. This way requires the additional analysis of a test generator work, and also additional hardware expenses for realization a modifying logic.

The most effective decision of the problem represents a method consisting in updating of an initial pseudo-random test patterns, offered in work [1]. The given method uses a condition of the test generator as entrance value for the logic function which modify specified bits of pseudo-random test patterns, that allows to receive the necessary deterministic test cube.

The basic lack of known methods [1 - 4] is the increase of hardware expenses at realization of the generator of test patterns, that in some cases is inadmissible for BIST schemes.

2.Technique of BIST, based on change of generated pseudo-random test patterns

Is shown, that with using LFSR as a test generator for pseudo-random patterns the majority of generated sequences are useless for faults detection in the combinational circuit [8] . At the same time, the detection of specific fault of circuits frequently is possible only with the deterministic test cubes, which are specified depending on the tested circuit [1]. As a rule, in pseudo-random test pattern number of bits requiring change is not large in comparison with common length of a test cube. In table 1 the statistical data on expected amount of bits in a pseudo-random patterns are given, which change is necessary for reception of the deterministic test cubes, depending on length of the test and amount of specified bits [1]. For example, with length of the test in 10000 patterns and with total specified bit equal 40, is necessary to change 7.83 bits in an

initial pseudo-random test patterns, that will allow to generate the necessary deterministic test cubes and, accordingly to ensure required fault coverage of tested circuits.

Table 1

Expected number of bits to be changed

Number of patterns	Number of specified bits						
	10	20	30	40	50	60	70
1000	0.02	2.78	6.09	9.54	13.32	17.17	21.11
10000	0.00	1.79	4.66	7.83	11.39	15.03	18.74
100000	0.00	0.90	3.53	6.50	9.65	13.19	16.64
1000000	0.00	0.05	2.54	5.21	8.29	11.52	14.89

The given data are submitted in work [1], in which the effective method of the polinomial analysis is offered. The method allows to synthesize an additional logic function for updating of generated pseudo-random test patterns for receiving necessary deterministic test cubes.

More attractive then known methods of making an additional logic for both generating pseudo-random and deterministic patterns is method of the analysis of primitive polinomial, allowing to prove a presence of the deterministic test cubes in all amount of generated pseudo-random patterns, and also, when such generation is impossibal to find a primitive polinomial which can generate needed deterministic test cubes. Such analysis of the generator is possible, because the quantity of primitive polinomials of a large value of degree is rather great and is defined as $L = \Ö (2m - 1) /m$, where Ö - Eyler function.

Thus, definition of presence of the needed test cubes with the specified bits in all amount of generated test patterns, and also, if necessary, find satisfying to the certain requirements primitive polinomial, is a solution of a problem of maximal fault coverage.

In the article the method of the primitive polinomial analysis for generator of pseudo-random test pattern is presented. The method allows to define an opportunity to generate of deterministic test patterns in all set of pseudo-random test cubes.. Described method is possible to use for circuits on BIST scheme design stage . The advantage of the offered method consists in absence of additional hardware overhead for generation of the deterministic test cubes.

3. Algorithm of the analysis polinomial for the test generator

The initial data for search of start conditions of the test generator ensuring presence of deterministic test paterns in all test cubes of generated pseudo-random test patterns is primitive polinomial $\varphi(x)$, length of a scan chain and set of the deterministic test patterns with specified bits.

The algorithm of the analysis for the test generator contains of the following stages.

1. The test generator provides producing periodically pseudo-random test patterns. In a consequence it, for each bits of a scan chain is possible to put in conformity both bits of pseudo-random patterns and bits of the deterministic test cubes.

2. On the basis of the analysis which has been made on a step 1, systems of equations for each deterministic test patterns are made. The given systems include equations for each specified bits of test patterns. The number of systems is defined by number of the deterministic patterns, and the number of the equations in each systems is defined by number of specified bits in the appropriate test cube.

3. Find primitive polinomial for generator of test patterns received by decimation of an initial pseudo-random sequence. Index of decimation should have a condition of mutual simplicity with a period of the test generator and the length of the scan chain.

4. Each bit of an initial sequence is put in conformity with bit of decimation sequence. In view of the received relations transform systems of equations received on a step 2 to systems for decimation sequence.

5. Calculate the coefficients for reception of the shifted copies of a M-sequence for shift value equal to numbers of bits of decimation test sequence rather which the systems of the equations on a step 4 are made.

6. The systems of the equations, received on a step 4, are transformed of rather start condition of the test generator. Coefficients in the equations are the appropriate value of coefficients for the shifted copies of the M-sequence.

7. Solve the systems of equations received on a step 6. The decisions of the systems of

equations are condition of the test generator in different steps of its work. It means, that they should differ from each other. The same decision of the systems will mean impossibility to generate needed deterministic test cubes. However, in practice the situation, when one deterministic sequence will include all specified bits of other patterns, will not take place. The given fact allows do not make the analysis of the decisions of the systems of equations.

8. If there is not decision of any system other primitive polinomial should be taken with equivalent or greater degree and make its analysis since a step 3 .

As it is clear from the algorithm, most work expenses and in a greater degree efficiency of the offered analysis is the step connected to evaluate of coefficients for evaluating of a shifted M-sequence. In the following unit the high speed alghoritm of getting coefficients of the shifted copies of a M-sequence is described.

4. Calculating of coefficients for getting the shifted copies of M-sequences

As was shown above, for the analysis of primitive polinomial of the test generator ensuring presence of deterministic test patterns in all set of generated pseudo-random cubes, it is necessary to evaluate coefficients for formation of the shifted copies of M-sequences.

There are many methods of the decision the problem distinguished on the efficiency [9]. The highest speed characterizes a method using the analytical analysis of test generator work.

The functioning of the generator of test sequences can be described as follows:

$$(1)\qquad a_1(k+1) = \sum_{i=1}^{m} \alpha_i * a_i(k);$$

$$(2)\qquad a_i(k+1) = a_{i-1}(k);\ i = \overline{2,m};\ k = \overline{0, 1, 2, \ldots},$$

where $a_i(k) \in \{0, 1\}$ – value of register bits on a k-step of work; $m = \deg \varphi(x)$ and $\varphi(x) = 1 \oplus \alpha_1 * x^1 \oplus \alpha_2 * x^2 \oplus \alpha_3 * x^3 \oplus \ldots \oplus \alpha_m * x^m$.

Using property of shift and addition of a M-sequence, it is possible to write down expression for definition of a M-sequence value shifted on any number of steps:

$$(3)\qquad a_1(k+u) = \sum_{i=1}^{m} \delta_i(u) * a_i(k);\ k = 0, 1, 2, \ldots$$

Where u - value of shift; $\delta_i(u) \in \{0, 1\}$- coefficients determining using of value an i-bit of the shift register for formation an element of a shifted M-sequence. From the point of view of realization the coefficient $\delta_i(u)$ set a connection topology of the bits of the shift register with additional XOR elements, on which output the shifted sequence is formed.

The method of definition coefficients $\delta_i(u)$ consists in definition of required value by the decision of some logic equations, which members are $\delta_i(h)$ and $\delta_i(s)$, where $h + s = u$. Let's assume, that the coefficients $\delta_i(h)$ and $\delta_i(u)$, allowing to receive M-sequence shifted on h and s steps, are known. Then according to (3) it is possible to write down

$$(4)\qquad a_1(k+h) = \sum_{n=1}^{m} \delta_n(h) * a_n(k)$$

And

$$(5)\qquad a_1(k+s) = \sum_{r=1}^{m} \delta_r(s) * a_r(k)$$

From (4) the validity of the next equality follows:

$$(6)\qquad a_1(k+h+s) = \sum_{i=1}^{m} \delta_n(h) * a_n(k+s)$$

Where $a_n(k+s)$ with the (5) is defined

$$(7)\quad a_i(k+s) = \sum_{p=1}^{m} \delta_p(s) * a_p(k),\ n = 1;$$

$$a_i(k+s) = \sum_{r=1}^{m-(n-1)} \delta_r(s) * a_{r+(n-1)}(k) \oplus \bigoplus_{q=1}^{n-1} \delta_{m-(n-1)+q}(s) * a_m(k-q),\ n = \overline{2,m}$$

With the (1) and (2) elements $a_m(k-q)$ can be submitted as follows:

$$(8)\quad a_m(k-q) = a_q(k) \oplus \sum_{c=1}^{m-q} \alpha_c * a_{q+c}(k) \oplus \sum_{d=1}^{q-1} \alpha_{m-q+d}(s) * a_m(k-q),\quad q = \overline{1,m-1}.$$

For reception expression (8) the fact, that for anyone m and primitive polinomial $\varphi(x)$ value of coefficient $\alpha_m = 1$ was used.

As a result of substitution of meanings $a_m(k-1)$, $a_m(k-2), \ldots, a_m(k-q+1)$ in (8) is received

$$(9)\qquad a_m(k-q) = \sum_{j=1}^{m} \beta_{j,q} * a_j(k),$$

Where $\beta_{j,q} \in \{0, 1\}$ is defined as:

$$(10)\quad \beta_{j,q} = 1, \text{ when } j = q = 1;$$

$$\beta_{j,q} = \sum_{u=1}^{q-1} \alpha_{m-u} * \beta_{j,q-u}, \text{ when } j = \overline{1, m-1},\ q = \overline{j+1, m-1};$$

$$\beta_{j,q} = \alpha_{j-1} \text{ when } j = \overline{2, m},\ q = 1;$$

$$\beta_{j,q} = \alpha_{j-q} \oplus \sum_{u=1}^{q=1} \alpha_{m-u} * \beta_{j,q-u}, \quad \text{when } j = \overline{1, m},\ q = \overline{2, j-1};$$

$$\beta_{j,q} = 1 \oplus \sum_{u=1}^{q-1} \alpha_{m-u} * \beta_{q,q-u}, \qquad \text{when } j = q = \overline{2, m-1}.$$

Consistently by substituting expression for $a_m(k-q)$ from (9) in (7) and $a_m(k+s)$ from (7) in (6) we shall receive

$$(11)\ \delta_j(h+s) = \sum_{v=1}^{j} \delta_v(h) * \delta_{j+1-v}(s) \oplus \oplus \sum_{q=1}^{m-1} \beta_{j,q} \sum_{c=q+1}^{m} \delta_c(h) * \delta_{m+q+1-c}(s), \quad j = 1, m$$

The expression (11) shows an opportunity of definition of coefficients $\delta_i(h+s)$ on set of meanings $\delta_i(h)$ and $\delta_i(h)$. Thus the given procedure is purely analytical.

With h = s and even j the first sum in expression (11) is equal to zero, when j is odd the first sum is $\delta_{(j+1)/2}$. In the case the ratio $\delta_v(h) * \delta_{j+1-v}(h) \oplus \delta_v(h) * \delta_{j+1-v}(h) = 0$ was taken.

With even m the system of the equations for $\delta_j(2h)$, including $\delta_j(2)$, is:

$$(12)\ \delta_j(2h) = \delta_{(j+1)/2}(h) \oplus \sum_{n=1}^{m/2} \beta_{j,2n-1}(h) * \delta_{m+1-n}(h), \quad j = 2k-1;$$

$$\delta_j(2h) = \sum_{n=1}^{m/2} \beta_{j,2n-1}(h) * \delta_{m/2+n}(h), \quad j = 2k,\ k = 1, 2, \ldots, i = \overline{1,m}$$

For odd m we have:

$$(13)\ \delta_j(2h) = \delta_{(j+1)/2}(h) \oplus \sum_{n=1}^{(m-1)/2} \beta_{j,2n}(h) * \delta_{(m+1)/2+n}(h), \quad j = 2k-1;$$

$$\delta_j(2h) = \sum_{n=1}^{(m-1)/2} \beta_{j,2n}(h) * \delta_{(m+1)/2+n}(h), \quad j = 2k,\ k = 1, 2, \ldots, i = \overline{1,m}$$

Let's note, that $\delta_j(1) = \alpha_j$, where $\alpha_j \in \{0, 1\}$- coefficients of primitive polinomial of initial M - sequence. Thus, on a basis (12) and (13), using meanings of coefficients $\delta_j(1)$ easily to calculate $\delta_j(1)$, further $\delta_j(1)$ and so on. If it is necessary to receive a copy of a M-sequence shifted on $u \neq 2^k$ of takts (k = 1, 2, 3, ...), in the beginning it is expedient to take advantage of expression (12) or (13), as having smaller computing complexity, and then to apply (11).

5.Example of the analysis a primitive polinomial of a test generator

The generator is given by polinomial $\varphi(x) = 1 \oplus x \oplus x^3$, length of a scan chain is equal to 5, the deterministic test patterns have the following kind:
1.{11xxx}; 2.{0xx1x}; 3. {11x01}.
The schema of the test generator and a scan chain is given in a fig. 1.

Fig. 1 The test generator and a scan chain used in the example

Step 1.According to the initial data, condition of a scan chain will be the following: 1. $\{a_4 a_3 a_2 a_1 a_0\}$; 2.$\{a_2 a_1 a_0 a_6 a_5\}$; 3. $\{a_0 a_6 a_5 a_4 a_3\}$; 4. $\{a_5 a_4 a_3 a_2 a_1\}$; 5. $\{a_3 a_2 a_1 a_0 a_6\}$; 6. $\{a_1 a_0 a_6 a_5 a_4\}$; 7. $\{a_6 a_5 a_4 a_3 a_2\}$.
Starting from various start condition, we shall receive identical patterns from seven test cubes. The choice of an initial condition will define only order of following of a test sequence in a set.

Step 2.For each deterministic patterns, being based on a condition of a scan chain, it is possible to write down the following equality:

1. {11xxx}: $a_{(4+k*5) \bmod 7} = 1$; $a_{(3+k*5) \bmod 7} = 1$;
2. {0xx1x}: $a_{(4+j*5) \bmod 7} = 0$; $a_{(1+j*5) \bmod 7} = 1$;
3. {11x01}: $a_{(4+j*5) \bmod 7} = 1$; $a_{(3+j*5) \bmod 7} = 1$; $a_{(1+j*5) \bmod 7} = 0$; $a_{(0+j*5) \bmod 7} = 1$.

Where $i \neq j \neq k \in \{0, 1, 2, \ldots, 6\}$.

Step 3.Let's examine a sequence received by decimation an initial sequence with decimation coefficient equal to length of a scan chain, in the case equal to 5: $b_i = a_{5*i \bmod 7}$. Primitive polinomial for decimation sequence has a kind: $\varphi(x) = 1 \oplus x^2 \oplus x^3$.

Step 4. $b_i = a_{5*i \bmod 7} \Rightarrow b_0 = a_0$, $b_1 = a_5$, $b_2 = a_3$, $b_3 = a_1$, $b_4 = a_6$, $b_5 = a_4$, $b_6 = a_2$.
The accordingly equality concerning a M-sequence received by decimation, will accept a kind:

1. {11xxx}: $b_{(5+k) \bmod 7} = 1$; $b_{(2+k) \bmod 7} = 1$;
2. {0xx1x}: $b_{(5+j) \bmod 7} = 0$; $b_{(3+j) \bmod 7} = 1$;
3. {11x01}: $b_{(5+j) \bmod 7} = 1$; $b_{(2+j) \bmod 7} = 1$; $b_{(3+j) \bmod 7} = 0$; $b_{(0+j) \bmod 7} = 1$.

Let's rewrite these equalities in view that any bit of a M-sequence can be received from an initial condition of the test generator by the coefficients of formation the shifted copies.

Step 5. The value of shifted coefficients of M-sequences for polinomial $\varphi(x) = 1 \oplus * x^2 \oplus * x^3$ for the appropriate meanings of shift have the following

meanings: 1.δ (0) = {1, 0, 0}; 2. δ (2) = {1, 1, 0}; 3. δ (3) = {1, 1, 1}; 4. δ (5) = {0, 0, 1}.
Step 6 and 7. Now it is possible to copy equality for specifióâ bits of a M-sequence of rather initial state of the generator:

$$0 * b_0 + 0 * b_1 + 1 * b_2 = 1$$
$$1 * b_0 + 1 * b_1 + 0 * b_2 = 1 \Rightarrow$$

$$b_0 = 0, b_1 = 1, b_2 = 1$$
$$b_{0+j} = 1, b_{1+j} = 0, b_{2+j} = 1$$

$$0 * b_{0+k} + 0 * b_{1+k} + 1 * b_{2+k} = 0$$
$$1 * b_{0+k} + 1 * b_{1+k} + 1 * b_{2+k} = 1 \Rightarrow k \neq 0$$

$$b_{0+k} = 0, b_{1+k} = 1, b_{2+k} = 0$$
$$b_{0+k} = 1, b_{1+k} = 0, b_{2+k} = 0$$

$$0 * b_{0+j} + 0 * b_{1+j} + 1 * b_{2+j} = 1$$
$$1 * b_{0+j} + 1 * b_{1+j} + 0 * b_{2+j} = 1$$
$$1 * b_{0+j} + 1 * b_{1+j} + 1 * b_{2+j} = 0$$
$$0 * b_{0+j} + 0 * b_{1+j} + 1 * b_{2+j} = 1 \Rightarrow j \neq k \neq 0$$

$$b_{0+j} = 0, b_{1+j} = 1, b_{2+j} = 1$$
$$b_{0+j} = 1, b_{1+j} = 0, b_{2+j} = 1$$

Step 8.Thus, all three systems have decisions. It means, that all needed deterministic test cubes can be generated. The concurrence of the decisions of the first and third system is a consequence that the first determined test patterns completely becomes covered by the third patterns, as was emphasized above, the probability of occurrence of such situation is small.

For example, if we shall take the decision of the first system as the start condition of the test generator, $b_0 = 0, b_1 = 1, b_2 = 1, \Rightarrow a_0 = b_0, a_1 = b_3, a_2 = b_6 \Rightarrow a_0 = 0, a_1 = 0, a_2 = 1$, according to the initial data, condition of a scan chain will be the following:
1. 0 1 1 1 0; 2. 1 1 0 0 1; 3. 0 0 1 0 1; 4. 1 0 1 1 1;
2. 5. 1 1 1 0 0; 6. 1 0 0 1 0; 7. 0 1 0 1 1.
As it is clear from the example needed deterministic patterns {11xxx} will be generated in the fifth case, the patterns {0xx1x} will be generated in first and in the seventh cases, and the patterns {11x01} in the second case.

6. Experimental results

The offered method of the analysis of primitive polinomial of the test generator of pseudo-random patterns was taken for a basis in realization of a software tools allowing to find primitive polimomial , ensuring an opportunity of generation of the deterministic test cubes. The developed software realizes two basic modes :
1. Mode of the analysis given primitive polinomial for an opportunity of generation of the needed deterministic test patterns with specified bits.
2. Mode of search optimum primitive polinomial $\varphi(x)$ with the minimal major degree m and minimal number of unzero factors α_i, ensuring of generation of the needed deterministic test cubes.

In table 2 data from work [4] are given about the deterministic test patterns necessary for achievement of the maximal fault coverage of the circuit. For example, the generator of pseudo-random test patterns is applied to the circuit s838 with a scan chain of length n = 66 bits on the basis of primitive polinomial of a degree m = 14. For achievement of the maximal fault coverage N = 159 deterministic test cubes with total specified bit equal S = 4415 are generated, and the maximal number of specified bit in one set is equal $s_{max} = 36$.

From tabel 2 follows, that quantity of specified bit in the deterministic test cubes makes an insignificant part in relation to common length of a scan chain. Experiments were made and established dependence between given deterministic patterns and number of specified bits in the patterns and ability of generating needed test cubes. The results of a various degree of polinomial are shown on fig.2. Length of a scan chain was 17 bits. The sets from N = 100 deterministic test cubes with random allocated specified bits equial s were used. The data given for polinomials with power m = 14 and m = 17.

On the basis that relative amount of specified bits in the deterministic pattrens insignificant (see tabel 2) given experimental data prove high efficiency of using this method of analysis for maintenance of the given maximal fault coverage of circuits.

7. Conclusion

In the article the new method of analysis of primitive polinomial for a test generator for BIST is presented. The method is allowing to define presence of the needed deterministic test patterns in all set of generated pseudo-random test cubes, and also, with impossibility of generation of the such cubes to find primitive polinomial, ensuring generation of needed deterministic patterns.

The analys of the experimental data, received with the developed software tools, allows to consider the offered method of the polynomial analysis as the method ensuring required fault coverage without increase hardware overhead by generation of deterministic test patterns, that is obvious advantage on compare with used nowadays methods of self-testing.

Table 2

Deterministic test patterns

Circuit	N	m	N	S	s_{max}
s641	54	14	8	167	22
s838	66	14	159	4415	36
s953	45	14	9	123	14
s5378	214	14	33	532	23

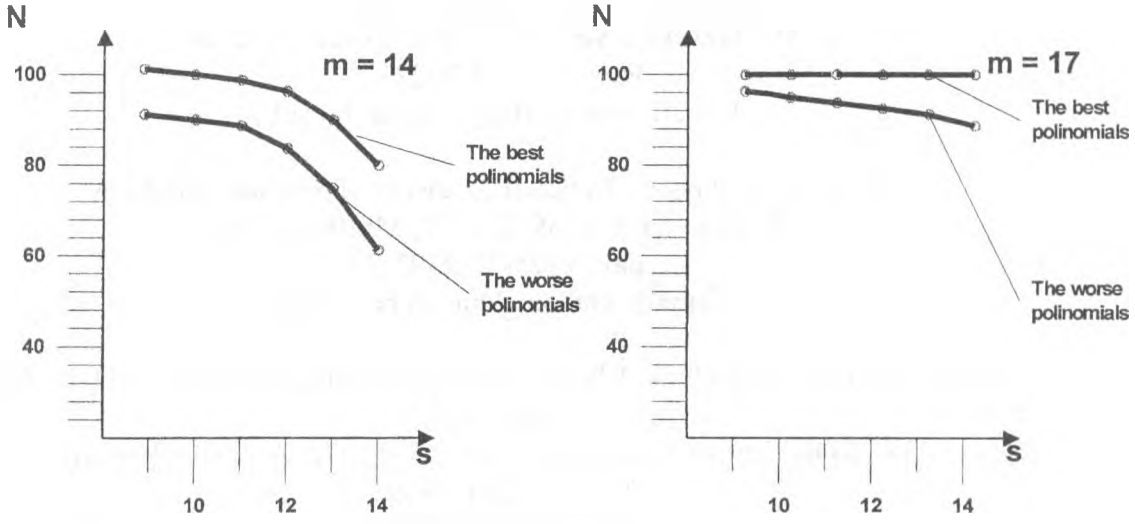


Fig. 2 The ability of generation deterministic patterns by polinomials with different power

References

1. **Hans-Joachim Wunderlich, Gundolf Kiefer. "Bit-Flipping BIST".** *IEEE Int. Test Conf.*, **1996. pp. 337-343.**
2. **Sybille Hellebrand, Hans-Joachim Wunderlich, Andre Hertwig. "Mixed-Mode BIST Using Embedded Processors".** *IEEE Int. Test Conf.*, **1996.**
3. **Sybille Hellebrand, Janusz Rajski, Stefen Tarnick, Srikanth Venkataraman, Bernard Courtois. "Built-In Test for Circuits with Scan Based on Reseeding of Multiple-Polynomial Linear Feedback Shift Register" .***IEEE Int. Test Conf.*, **1995. pp. 223-233**
4. **Sybille Hellebrand, Birgit Reeb, Stefen Tarnick, Hans-Joachim Wunderlich. "Pattern Generation for Deterministic BIST Scheme" .** *IEEE Int. Test Conf.*, **1995. pp. 88-94.**
5. **Stephen Pateras, Janusz Rajski. "Cube-Containsed Random Patterns and their Application to the Complete Testing of Synthesized Multi-level Circuits".** *IEEE Int. Test Conf.*, **1991. pp. 473-482.**
6. **Christophe Fagot, Patrick Girard, Cristian Landrault. "A Novel Approach for Logic BIST Based on Machine Learning ".** *IEEE Int. On-Line Testing Workshop*, **1997. pp. 170-174.**
7. **Gundolf Kiefer, Hans-Joachim Wunderlich. "Using BIST Control for Pattern Generation ".***IEEE Int. Test Conf.*, **1997. pp. 347-355.**
8. **Nur A.Touba, Edward J. McCluskey." Syntethis of Mapping Logic for Generating Transformed Pseudo-Random Patterns for BIST".** *IEEE Int. Test Conf.*, **1995.**
9. **V.N. Yarmolik , S.N. Demidenko.** *Generating and Using Pseudo-Random Signals in Test Systems* **. Science and Engineering Publisher, Minsk,1986.**

Computer Modeling the Excitonic Reflection and Photoluminescence Spectra of GaN Epitaxial Layers

Yu. Rakovich, N.P. Tarasjuk, A.A. Gladyshchuk*,
E.V. Lucenko**, G.P. Yablonskii**, M. Heuken***, K. Heime****

*Department of Physics, Brest Polytechnic Institute,
Moskowskaja Str., 267, 224017, Brest, Belarus
Fax: +375-16-2422127
E-mail: Physics@brpi.belpak.brest.by

**Institute of Physics, National Academy of Science of Belarus,
F. Skaryna Ave. 68, 220072, Minsk, Belarus
Fax: +375-17-2393131
E-mail: Yablon@dragon.bas-net.by

***Institut für Halbleitertechnik, RWTH Aachen, Templegraben 55, D-52056, Aachen, Germany
AIXTRON GmbH, Kackert Str. 15-17, D-52072, Aachen, Germany
Fax: +49-241-890940
E-mail: Heu@aixtron.com

****Institut für Halbleitertechnik, RWTH Aachen, Templegraben 55, D-52056, Aachen, Germany
Fax: +49-241-8888 199
E-mail: mailbox@iht.rwth-aachen.de

Abstract

Photoluminescence (PL) and reflection excitonic spectra of GaN single layer grown on sapphire substrate by MOVPE were modeled with aim to estimate a basic parameters of free A- and B- excitons. The calculations were performed in the frame of two-oscillator model for dielectric function ε(E). Three layered model of crystal was used for fitting of reflection spectrum which was measured at T=80K. In this way the dead layer thickness d = 6 nm, resonance energies E_A= 3.4916 eV and E_B = 3.5008 eV as well as the broadening parameters Γ_A = 5.27 meV and Γ_B = 7.14 meV of the free excitons were obtained. These parameters were used then for fitting of PL spectra in assumption of the thermal equilibrium for excitons taking into account the self-absorption of resonance emission. The values of diffusion coefficients D_A = 0.3 cm^2/s, D_B = 0.1 cm^2/s and exciton lifetimes τ_A = 37 ps, τ_B = 17 ps were estimated.

1. Introduction

GaN and related nitrides have attracted strong interest for application such as blue-UV light emitting diodes, blue-UV laser diodes, video displays, full color TV systems, high - density optical memory systems for computer networks. The technological development of GaN based optoelectronic devices makes the investigation of the properties of an excitonic states of great practical importance. Fundamentally the analysis of the line shape of free exciton reflection and PL spectra can provide a way for determination of energetic structure of semiconductors and estimation of basic parameters of radiative recombination processes. However it meets with a serious difficulties practically because of both observed PL and reflection spectra are usually transformed to a variable degree. It is well known that resonance reflectance in region of exciton transitions is very sensitive to state of the surface of the crystal. Various treatments of crystals (illumination, etching, electron and ion

bombardment, field effect, temperature variation, ets.) may alter radically the exciton line shape. The phenomena has been interpreted initially in terms of free exciton layer (FEL) model which was introduced to account for the exciton repulsion from surface arising from the image potential [1]. The interference between light waves reflected at the outer and inner boundaries of the layer proved to have a important role on the reflectivity spectra. We used FEL model in present work to obtain theoretical reflectance curve for comparison with experiment.

It is not such widely known , that the FEL approach can be applied also to solution of the opposite problem: description of excitonic emission from crystal under photogeneration [2]. The experimentally observed PL lineshape in resonance region of free excitons is determined greatly by transmission coefficient of light originating from crystal volume [3, 4]. Solving of this problem can be very useful for interpretation not only excitonic PL spectra but also absorption spectra as well as spectra of Raman and Brillouin scattering [5]. In present work we calculated the transmission coefficient in terms of the PL light interference at the FEL boundaries. Self-absorption of resonance radiation was also taken into account. The results of calculations was used then for fitting of PL spectra of the free A- and B- excitons in the GaN epitaxial layer.

2. Growth and experimental procedure

The 2 μm thick undoped single GaN layer was grown on c-plane sapphire substrate by MOVPE in an AIXTRON reactor. The layer shows n-type conductivity with a free electron concentration $5 \cdot 10^{17}$ cm^{-3}.

Photoluminescence was excited using He-Cd laser with λ_{exc} = 325 nm. The excitation intensity I_{exc} was 0.5 W/cm^2. The PL and reflection spectra were registered using a diffraction monochromator and a photomultiplier and then re-calculated to include the spectral sensitivity distribution of the monochromator - photomultiplier system. The spectral resolution was less than 0.3 meV (20-80K) and <1 meV above 80K.

3. Theoretical calculations

3.1. Reflection spectra

Optical properties of a medium with complex refractive index $\tilde{n} = n - i\kappa$ are determined in investigated spectral region by an excitonic oscillators with following parameters: resonance frequency ω_0 (or energy E_0), the broadening (or decay) parameter Γ, the exciton polarizability $4\pi\alpha$, the background dielectric constant ε_b [5].

We assume a model of two coupled oscillators for the dielectric function near A and B free exciton transition energies:

$$\tilde{\varepsilon}(\omega) = \varepsilon_b + \frac{4\pi\alpha_A\omega_{0A}^2}{\omega_{0A}^2 - \omega^2 - i\omega\Gamma_A} + \frac{4\pi\alpha_B\omega_{0B}^2}{\omega_{0B}^2 - \omega^2 - i\omega\Gamma_B}, \quad (1)$$

where indexes A and B correspond to the A and B exciton states respectively.

To calculate the reflection spectra R(ω) taking into account the interference in FEL we used the methods described in [1] and corresponding formulas:

$$R(\omega) = \tilde{r}(\omega)\cdot\tilde{r}^*(\omega),$$

$$\tilde{r}(\omega) = \frac{r_{12} + \tilde{r}_{23}(\omega)e^{2i\Theta}}{1 + r_{12}\tilde{r}_{23}(\omega)e^{2i\Theta}}, \quad (2)$$

with

$$\Theta = 2\pi d / \lambda_m,$$

where λ_m is the light wavelength within the medium, d is the FEL thickness, and

$$r_{12} = \left(1 - \sqrt{\varepsilon_b}\right)\Big/\left(1 + \sqrt{\varepsilon_b}\right);$$

$$\tilde{r}_{23}(\omega) = \left(\sqrt{\varepsilon_b} - \tilde{n}(\omega)\right)\Big/\left(\sqrt{\varepsilon_b} + \tilde{n}(\omega)\right);$$

$$\tilde{n} = \sqrt{\varepsilon(\omega)}.$$

The complicating factors such as the possible inhomogeneous broadening, the frequency dependence of Γ, the temperature dependence of polarizability and background dielectric constant, as well as the anisotropy of ε_b, were neglected, because the inaccuracy caused by these factors is small [6].

3.2. Luminescence spectra

The evaluation of excitation levels, used in our experiments, shows that at created concentration of carries about 10^{12} cm^{-3} at I_{exc}=0.5 W/cm^2 estimated by generation rate, any degeneracy does not take place. Thus, the radiation line shape is described in steady-state case in one-coordinate approximation by the following equation [7]:

$$I_{PL}(\omega) = \rho(\omega)\left(1 - R'(\omega)\right) \times \int_0^{\hbar} n_{ex}(x)\exp\left(-k(\omega)x\right)dx, \quad (3)$$

where h is the epilayer thickness, $\rho(\omega)$ is the probability per unit time of emitting a photon with the frequency ω by exciton annihilation, $R'(\omega)$ is the reflectivity of light with the frequency ω incident on the surface from the epi-layer, $k(\omega)$ is the absorption coefficient of GaN, $n_{ex}(x)$ is the excitonic concentration. The probability of exciton-photon transition $\rho(\omega)$ was calculated as the probability of the process reciprocal to the absorption of photons [8].

Under the conditions of quasi-equilibrium between the free carrier gas and excitons, the value of $n_{ex}(x)$ can be obtained [7]:

$$n_{ex}(x) = \frac{I_{exc}\,k\,\tau}{L^2 - k^2 - 1} \times \left[\frac{k - \frac{S}{D_{ex}}}{\frac{1}{L} - \frac{S}{D_{ex}}} \exp\left(-\frac{x}{L}\right) - \exp(-kx) \right], \quad (4)$$

where S is the surface recombination rate, k is the absorption coefficient on the pump wavelength, τ is the total life time of exciton, D_{ex} and L are diffusion coefficient and the diffusion length of the exciton respectively.

3.3. Fitting procedure

The optimizing of parameters was carried out by the coordinate-wise descent method [9] with parabolic interpolation. The function

$$F = \sum_{j=1}^{N} \left(Y_{calc_j}(\omega) - Y_{exp_j}(\omega) \right)^2 + \beta \sum_{j=1}^{N} \left[\frac{Y_{calc_j}(\omega) - Y_{exp_j}(\omega)}{\Delta\omega} \right]^2, \quad (5)$$

where N is the number of the experimental points, $Y_{exp\ j}$ and $Y_{calc\ j}$ are experimental and calculated values of the reflectance or the PL intensity, respectively, β is the weight parameter of the curve shape, was used as an "aim function". The first term in (5) characterizes the fitting of absolute values of the reflectivity or the PL intensity, the second one represents the rate of their changes, i.e. the curve shape.

At initial stage of optimization the curve shape had the greater weight (the greater β parameter), then, after each 25 iterations, the parameter α decreased in 0.68 times (the golden mean), which allowed to adjust the absolute values of Y in given spectral region after fitting the shape of the curve.

4. Results and discussion

The major difficulty from standpoint of quantitative analysis of free excitons line-shape of GaN epilayers is the overlapping of PL lines which is observed generally over all temperature region. The PL spectra at low temperature (Fig1.a) exhibit a low intensity PL band $A_{n=1}$ (3.4946 eV) attributed to the free exciton annihilation and a line I_2 bound to the neutral donors , which in reality includes the set of at least three overlapped lines [10].

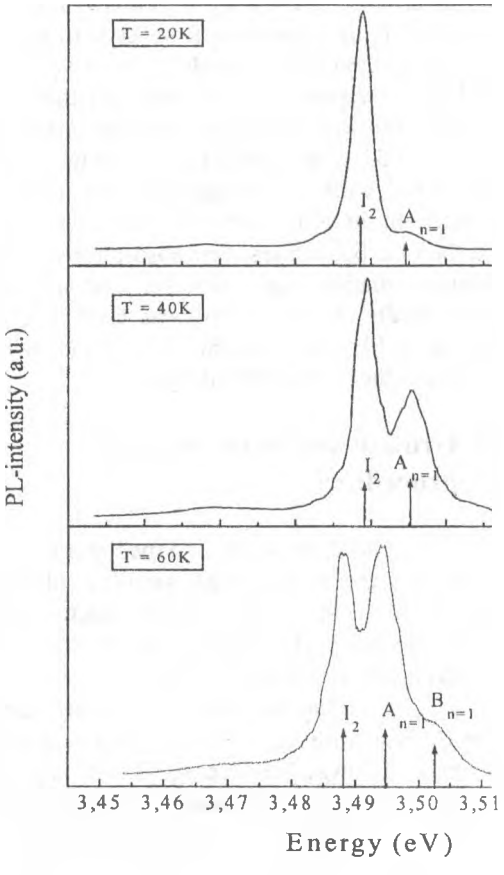


Fig.1. PL spectra of GaN epitaxial layer in the exciton region.

The dominance of bound exciton band in observed PL spectra together with low intensity of $A_{n=1}$ lines gives no way to reliable analysis of free exciton lineshape. Fortunately an increase of temperature leads to an efficient quenching of the bound exciton line due to of small activation energy (Fig.1). The I_2 band disappear gradually from the spectrum above

60-80K. Therefore to avoid problems with overlapping of luminescence lines we used for analysis the PL spectra measured at T=80K. At this temperature, lines related to the annihilation of the free exciton states become dominating in the PL spectra (Fig.2,a).

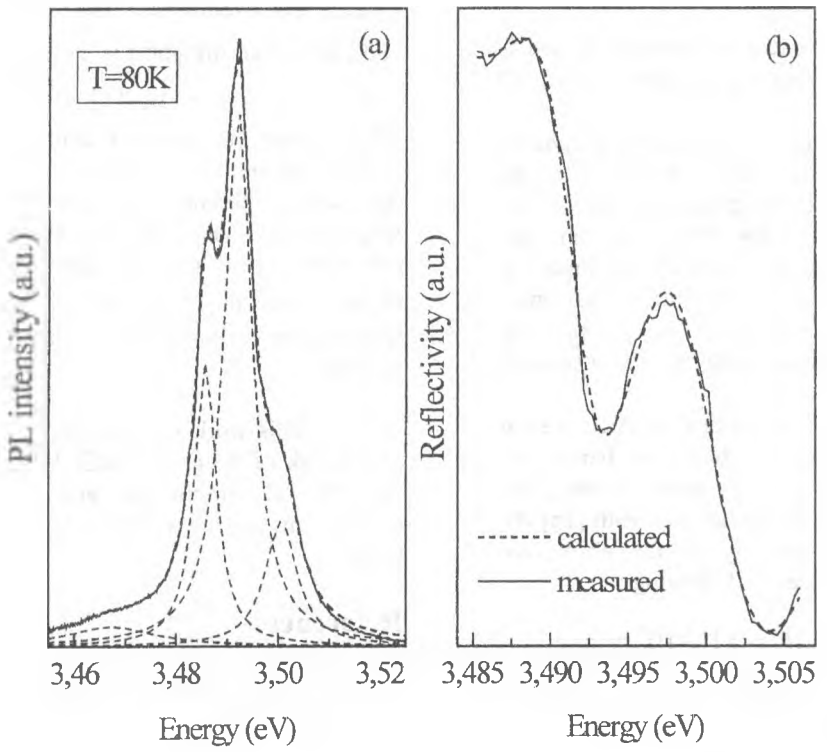


Fig.2. Experimental (solid curves) and calculated (dashed curves) PL (a) and reflection (b) spectra of a GaN sample for T=80K.

An additional reason for our choice was an assumption that the spatial dispersion (SD) effect can be eliminated from consideration at this temperature. Including of the SD effect (it means a dependence of dielectric function $\varepsilon(\omega)$ on wave vector $\bar{k}$) leads to substitution of the additional term $\hbar\bar{k} / M^*$ (where M* is the effective mass of exciton) in to denominator of equation (1). Though the energy position of the exciton resonance E_0 does not depend on whether we include or neglect the SD effect in the calculations [2], the last substitution results in a serious complication of computation procedure.

The temperature region where the SD effect should be taken into account, can be estimated by calculation of the critical broadening parameter Γ_{cr}. The transition to classical dispersion model is marked by the beginning of the strong temperature broadening of the exciton absorption band (or, what is the same, of the Γ parameter) [11]. It means that Γ parameter must be greater than the Γ_{cr} at fixed temperature.

Calculation of Γ_{cr} was performed using formula [11]:

$$\Gamma_{cr} = \sqrt{\frac{\hbar\omega_T^2 \varepsilon_b \Delta_{LT}}{2M^* c^2}},$$

where $\Delta_{LT} = \omega_0\left(\sqrt{1+\frac{4\pi\alpha}{\varepsilon_b}} - 1\right)$ is the longitudinal-transversal splitting, and $\omega_0 = \omega_T$ was assumed. The calculation gives Γ_{crA}=4.8 meV and Γ_{crB}=6.8 meV for A- and B- exciton respectively, if we admit $M^* = 0.97m_0 = 0.49567\times10^6$ eV [13], $4\pi\alpha_A=2.7\times10^{-3}$ [14], $4\pi\alpha_B=3.1\times10^{-3}$ [14] and ε_b=5.2 [14].

The fitting of experimental reflection spectrum was performed using the procedures described above to check the admissibility of neglecting of SD effect. The experimental and calculated reflection spectra are given in Fig.2 demonstrating an excellent agreement among them. The next parameters of exciton transitions of epitaxial layer GaN were evaluated:

$E_{0A}= \hbar\omega_{0A}$ =3.4912 eV,

Γ_A = 5.27 meV,

$f_A = 0.0043$,
Δ_{LTA} =9.1 meV (for $A_{n=1}$ exciton);

$E_{0B} = 3.5008$ eV,
$\Gamma_B = 7.14$ meV,
$f_B = 0.0049$,
Δ_{LTB} =1.0 meV (for $B_{n=1}$ exciton);
d = 6.1 nm.
The values of the oscillator strength f_A and f_B were calculated using the equation $f = \alpha / 0.2$ [15].

The values of broadening parameter obtained from the "best fitting" of the experimental reflection spectra exceed the corresponding critical ones. Thus, we can conclude that SD effect does not play an important role at indicated temperature. The most probable cause of that is a possible inhomogeneous broadening of the excitonic lines.

Integrated absorption coefficient K of A- and B- free exciton absorption bands was also estimated using the obtained values of the oscillator strength. In accordance with [16] the total integration intensity of the absorption peak can be expressed as following :

$$K = \int k(E)dE = 110 \cdot 10^6 \frac{f}{n\Omega_0}$$

where Ω_0 -is the volume of GaN lattice cell (in A^3), n is the refractive index.

Using of our data we evaluated integrated absorption coefficient to be $K_A = 4.459 \times 10^3$ eV/cm and $K_B = 4.149 \times 10^3$ eV/cm for A - and B- exciton bands respectively.

To calculate $I_{PL}(E)$, the parameters E_0, Γ and d obtained from results of the fitting of the experimental reflection spectra were used. The reflection coefficient of excitonic radiation falling on the surface from volume of the epilayer $R'(\omega)$ was calculated at fixing of these parameters. Formulas analogous to (2) was used, where r_{12} is replaced by the complex reflectivity at inner plane of FEL $\tilde{r}_{12}$ and instead of $\tilde{r}_{23}$ the real reflectivity at external surface of sample r_{23} is used. The absorption coefficient k(ω) was calculated as $\frac{2\omega \operatorname{Im} \tilde{\varepsilon}(\omega)}{c}$, where c is velocity of light.

Parameters $I_{exc} = 1.6 \times 10^{18}$ photon$\times$cm^{-2}s^{-1}, k = 10^6 cm^{-3} and S=100 cm/s were used. The PL intensity calculations were performed separately for each exciton band and then results were summed. The simple Lorentz model was used for modeling of emission bands placing in low energy side from A-exciton peak position. Only two free exciton parameters, namely the exciton diffusion coefficient and the exciton lifetime were varied during of fitting procedure described above. The the experimental and calculated PL spectra are given in fig.2,a.

By fitting the experimental PL spectra, the exciton diffusion coefficient D_{ex} and the exciton lifetime τ were estimated: D_{ex}^A = 0.3 cm^2/s, τ_A = 36 ps, D_{ex}^A = 0.1 cm^2/s, τ_B = 17 ps, giving the values of diffusion length L_A = 0.033 μm and L_B = 0.013 μm. As one can see the exciton lifetimes and values of diffusion length in our sample is very short, such that diffusion processes can be neglected. It should be noted that the value of τ is confirmed by time-resolve experiments in the picosecond regime.

This work was carried out within the framework of the project BELARUS-INTAS - 98-0995 "Recombination and compensation mechanisms in GaN epitaxial layers and structures".

References

1. J.J. Hopfield, D.G. Thomas. Theoretical and Experimental Effects of Spatial Dispersion on the Optical Properties of Crystals // Phys. Rev. **32** (1963) 563-572
2. A.L. Gurskii, Yu.P. Rakovich, M.M. Karpuk, A.A. Gladyshchuk, G.P. Yablonskii, H. Hamadeh, W. Taudt, M. Heuken. Structure of free exciton luminescence spectra in heteroepitaxial ZnSe/GaAs // J. Cryst. Growth **184/185** (1998) 1100-1104
3. S.A. Permogorov, A.V. Sel'kin Transmission of emission through crystal surface in region of exciton resonance taking into account the spatial dispersion effect. // Fiz. Tverd. Tela **15** (1973) 3025-3028.
4. Yu.P. Rakovich, G.P. Yablonskii, A.A. Gladyshchuk, A.S. Smal Influence of Surface Electric Field on exciton Photoluminescence of CdS // Phys. stat. sol. (b) **189** (1995) 247-256.
5. C.F. Klingshirn Semiconductor Optics. Springer-Verlag, Berlin Heidelberg New York, 1997, P.249
6. KA. Dmitrienko, L.V. Taranenko, S.G. Shevel, A.V. Marinichenko. Temperature dependence (4.2-300K) of resonance energies of exciton transitions in

single crystals A^2B^6 // Fiz. and Techn. Poluprov. **19** (1985) 788-794.
7. V.M. Agranovich. Theory of Excitons. Moscow, Nauka Press, 1982. P. 313-319
8. V.V. Travnikov, V.V. Krivolapchuk. Diffusion of excitons and self-absorption of resonance emission // Sov. Fiz. Tverd. Tela, **24** (1982) 961-969.
9. Yu.P.Boglaev. Numerical mathematics and programming. Moscow, Wysshaya Shkola Press, 1990, p.367.
10. G.P. Yablonskii, A.L. Gurskii, E.V. Lutsenko, I.P. Marko, B. Schineller, A. Guttzeit, O. Schön, M. Heuken, K. Heime, R. Beccard, D. Schmitz, H. Juergensen Optical properties and recombination mechanisms in GaN and GaN:Mg growth by metalorganic vapour phase epitaxy // J. Electr. Mat. **27** (1998) 222-228
11. M.I. Strashnikova, V.V.Cherny. Temperature dependence of broadening parameter of 1A excitons in CdS crystal // Sov.Fiz. Tverd. Tela, **33** (1991) 1134.
12. S.A.Permogorov. Resonant Spectra of free excitons. In: Physics of II-VI compounds, ed by A.N.Georgobiani and M.K.Sheinkman, Moscow, Nauka Press, 1986, p. 146-183.
13. D. Volm, K. Oettinger, T. Streibl, D. Kovalev, M. Benchorin, J. Diener, B.K. Meyer, J. Majewski, L. Eckey, A. Hoffmann, H. Amano, I. Akasaki, K. Hiramatsu, T. Detchprohm Exciton Fine-Structure in Undoped GaN Epitaxial-Films // Phys. Rev. (B) **53** (1996) 16543-16550
14. K.P. Korona, R. Stepniewski, J.M. Baranowski Dielectric function theory calculations of polaritons in GaN // Acta Physica Polonica A. **92** (1997) 867-870.
15. D.K. Nelson, Yu.V. Melnik, A.V. Selkin, M.A. Jakobson, V.A. Dmitriev, K. Irvin, K.X. Karter Optical investigation of exciton spectra in epitaxial GaN films // Sov. Fiz. Tverd. Tela, **38** (1996) 822-826.
16. R.S. Knox Theory of excitons. Moscow, Mir, 1966. P.124

Decomposition Approach to Minimize a Class of Superposition of Recurrent Monotone Functions on a Set of Parameterized Paths in Digraph

Nikolai Guschinsky and Genrikh Levin
Institute of Engineering Cybernetics, National Academy of Sciences of Belarus,
Surganova str., 6, 220012 Minsk, Belarus

Abstract

A problem of finding optimal parameterized path in digraph is considered. A parameterized path is a path each arc of which is assigned a parameter from a given set. A superposition of recurrent-monotone functions is accepted as an objective function where one of the functions is defined by using max operation. A two-level decomposition scheme for solving an initial problem is proposed. Methods for solving the obtained after decomposition subproblems are developed.

1. Introduction

Problems of finding optimal paths often arise in design and analysis of various networks. Shortest path problem is one of the most well-known such problems. A lot of papers is devoted to this problem (see, for instance, bibliography in [1, 2]). Last decades formulations of these problems with complex objective functions are intensively studied. Sometimes such problems arise when reducing multicriteria problems to a problem with one so-called generalized criterion which is in fact a superposition of partial criteria. This paper focuses on a class of such problems.

In the section 2, the considered problem is formulated. In the section 3, a twolevel decomposition scheme for solving the initial problem is proposed. In the following sections, methods for solving subproblems obtained after decomposition are developed taking into account their features.

2. Statement of the problem

Let $G=(V,E)$ be a finite directed graph with distinguished vertices s and t without multiple arcs. Each arc $(v,p)\in E$ is assigned a set Γ_{vp} and a vector-function $c_{vp}(\alpha)=(c^1_{vp}(\alpha), c^2_{vp}(\alpha), c^3_{vp}(\alpha))$ where $\alpha \in \Gamma_{vp}$ and $c^r_{vp}(\alpha)$, $r=1,2,3$, are non-negative real-valued functions.

A pair $x=(w,\gamma)$ is a parameterized path in graph G, if $w=(i_0,\mathrm{K},i_l)$ is a path in graph G and $\gamma=(\gamma_1,\mathrm{K},\gamma_l)\in\Gamma(w)=\prod_{k=1}^{l}\Gamma_{i_{k-1}i_k}$.

On a set X_v of parameterized paths in graph G from the vertex s to a vertex $v\in V$, the following functions are defined:

$$f^1(x)=\max\{c^1_{i_{k-1}i_k}(\gamma_k)\mid 1\le k\le l\},$$

$$f^r(x)=\sum_{k=1}^{l} c^r_{i_{k-1}i_k}(\gamma_k), r=2,3.$$

An initial problem **A** is to find a parameterized path $x^*=(w^*,\gamma^*)\in X=X_t$, which minimizes a function $g(x)=f^1(x)\cdot f^2(x)+f^3(x)$.

3. Decomposition scheme

For solving the problem **A**, the following decomposition scheme can be used. It is a concrete definition of a general scheme for solving similar class of problems [3, 4, 5].

Let us introduce a set $Y\subset R$ such that

$$Y\cap\{f^1(x)\mid x\in X^*\}\neq\varnothing$$

and

$$[\min\{f^1(x)\mid x\in X\},\max\{f^1(x)\mid x\in X\}]\supseteq Y$$

where X^* is a set of solutions of the problem **A**, and a function $g^0(x,y)=y\cdot f^2(x)+f^3(x)$ which is defined on $X\times Y$.

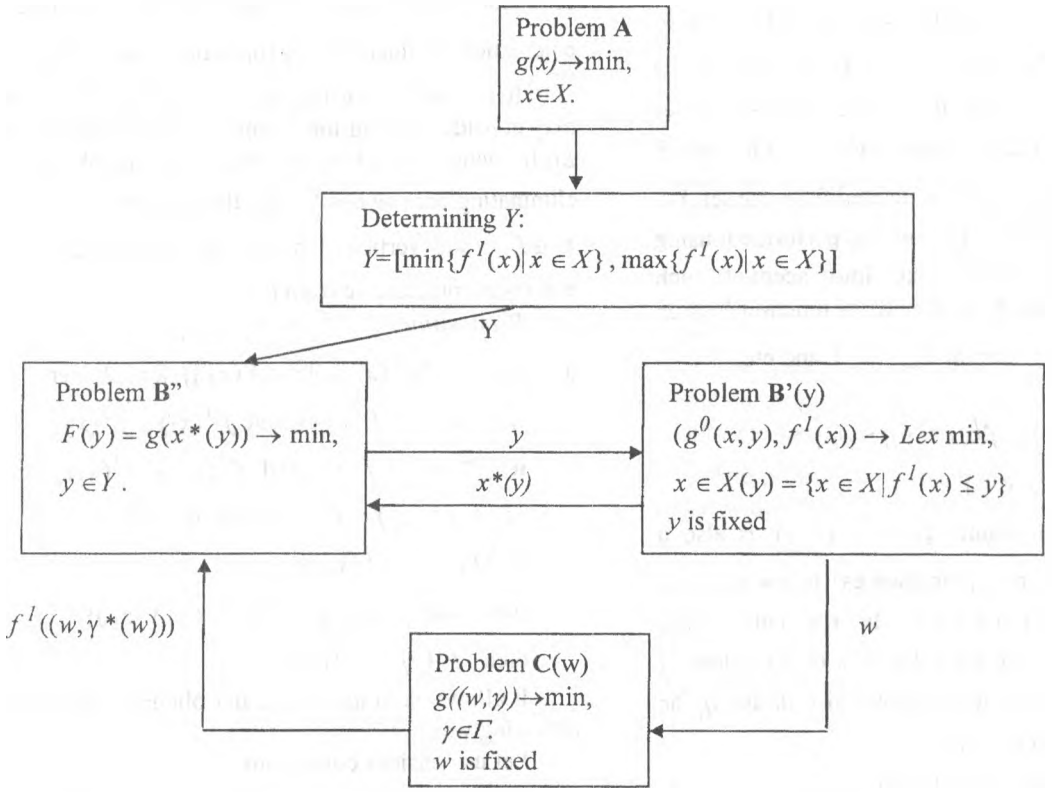


Fig.1. Decomposition scheme for solving the initial problem **A**

At a lower level, for fixed value of parameter $y \in Y$ we solve (possible approximate to a criterion $f^1(x)$) local problems **B'**(y) of lexicographic minimization

$$(g^0(x,y), f^1(x)) \to Lex \min,$$

$$x \in X(y) = \{x \in X | f^1(x) \le y\}$$

and at upper level, we solve problem **B"**:

$$F(y) = g(x^*(y)) \to \min,$$

$$y \in Y,$$

where x*(y) is an obtained solution of the problem **B'**(y),

Solution $x^*(y)=(w,\gamma)$ can be "improved" after solving an auxiliary problem **C**(w):

$$g((w,\gamma)) \to \min,$$

$$\gamma \in \Gamma.$$

Diagram of this procedure is shown in Fig. 1.

Proposition 1. If y is a ε-approximate solution of the problem **B"**, then x*(y) is a ε-approximate solution of the problem **A**.

It should be noted that if sets Γ_{vp} are finite for all $(v,p) \in E$, then the problem **A** is polynomially (with regard to dimensions of graph G) solvable.

4. Solving the problem B"

For solving the problem **B"** we propose to use the following modification of "branch and bound method" which takes into account non-unimodality of function F(y) on Y.

To a current step of an algorithm, a value y^0 and a current record $F^0 = F(y^0)$ of function $F(y)$ on Y are known as well as from the initial set Y a set Y^0 is extracted such that $\min\{F(y)|y \in Y^0 \cup \{y^0\}\} - F^* \le \varepsilon$ where $F^* = \min\{F(y)|y \in Y\}$ and ε is a required accuracy (with regard to objective function) for solution of problem **A**. The set Y^0 is divided into subsets Y_i contained in disjoint intervals (y_i^-, y_i^+).

At the current step, a value y' is chosen in a set Y_i for which problem **B'**(y') is solved and parameterized path $x^*(y')=(w^*(y'), \gamma^*(y'))$ is found. The problem **C**(w) for $w=w^*(y')$ is solved (if

necessary). It results in correction of values y^0 and F^0, replacing the set Y_i by its subsets $Y_{i1} = \{y \in Y_i \mid y < y'\}$ and $Y_{i2} = \{y \in Y_i \mid y > y'\}$. After that for each subset Y_i of Y^0, a value z_i is defined in such a way that either $F(y) \geq F^0 - \varepsilon$ for $z_i \leq y \leq y_i^+$ or there exists $y' < z_i$ for which $F(y') \leq F(y)$. Subset $[z_i, y_i^+]$ is deleted from the set Y_i.

A choice of next Y_i can be performed using various heuristics which take into account such characteristics of sets Y_i as bounds of function $F(y)$ on the set Y_i, length of interval (y_i^-, y_i^+) and etc.

Since

$$\min\{F(y) \mid y \in Y_i, f^1(x^*(y)) > y_i^-\} \geq$$

$$\min\{g^0(x,y) \mid y \in Y_i\}$$

then either a lower bound q_i of $g^0(x,y)$ is also a lower bound of $F(y)$ on Y_i or there exists $z < y_i^-$ such that $F(z) \leq \min\{F(y) \mid y \in Y_i\}$. In the latter case, eliminating Y_i from the set Y does not result in loss of solution to the problem **B**″. It allows also to use q_i as lower bound of $F(y)$ on Y_i.

In particular, we can consider

$$q_i = \max[g' - (y' - y_i^-)\beta_i^2, g' y_i^- / y' + \beta_i^3 (1 - y_i^- / y)]$$

where β_i^2 is an upper bound of function $f^2(x)$ and β_i^3 is a lower bound of function $f^3(x)$ on the set $\{x \in X_i \mid y_i^- < f^1(x) < y_i^+\}$. The eliminated set $[z_i, y_i^+]$ can be defined as

$$z_i = \min[y' - (g' - F^0)/\beta_i^2, y'(g' - F^0)/(g' - \beta_i^3)].$$

It should be noted that if the bound β_i^3 is accessible then its use is preferable in comparison with β_i^2.

5. Solving the problem B′(y)

A general scheme for solving the problem **A** requires to solve a sequence of problem **B**′(y) for some sequence $\{y_t\}$ of parameters $y \in Y$, which is generated during solving the problem **B**″. It results in a development of special methods for solving problems **B**′(y) which take into account their parametric properties. Methods proposed hereafter allow to use for solving the current problem **B**′(y) data which have already been obtained during solving problems **B**′(y) for the nearest lesser and greater values to y from the sequence $\{y_t\}$. Such approach is very effective if calculation of functions $c_{vp}^r(o)$ is time consuming.

It is easy to see that for the problem **B**′(y) one may consider instead the graph G a graph $G(y)=(V(y), E(y))$ which is obtained from the graph G by eliminating arcs $(v,p) \in E$ such that $c_{vp}^l(\alpha) > y$ for all $\alpha \in \Gamma_{vp}$ and vertices that are not accessible from s and counteraccessible from t.

Proposition 2. If $z_1, z_2 \in \{y_t\}$, $z_1 < z_2$ and $x_k = \arg\min\{g^0(x, z_k) \mid x \in X(z_k)\}$, $k=1,2$, then

i) $f^2(x_1) \geq f^2(x_2)$ and $f^3(x_1) \leq f^3(x_2)$;

ii) $f^2(x_1) = f^2(x_2)$ iff $f^3(x_1) = f^3(x_2)$;

iii) if $f^2(x_1) = f^2(x_2)$ and for $k=1,2$,

$$(g^0(x, z_k), f^1(x_k)) =$$

$$Lex\min\{(g^0(x, z_k), f^1(x)) \mid x \in X(z_k)\},$$

then $f^1(x_1) = f^1(x_2)$.

It allows us to use the earlier obtained data in the following way.

Let us consider collections

$H^n(y) = \{h_p^n(y) \mid p \in V(y)\}$ of vectors

$h_p^n(y) = (h_p^{n,1}(y), h_p^{n,2}(y), h_p^{n,3}(y))$, $n=1,2,3$

such that $h_v^n(y) = (0,0,0)$ and for all $p \in V(y) \setminus \{s\}$ the following recurrent relationships are satisfied:

$$h_p^n(y) = \arg lex\min\{(y \cdot \lambda^2 + \lambda^3, \lambda^1) \mid (\lambda^1, \lambda^2, \lambda^3) \in Z_p^n(y)\} \quad (1)$$

moreover if $h_p^n(y) \neq h_p^n(y_p^n(y))$, then there exists a parameterized path

$$x_p^n(H_p^n(y)) = ((v = i_0, \ldots, i_l), (\gamma_1, \ldots, \gamma_l))$$

such that $h_v^n(y) = h_v^n(y_v^n(y))$ and for $k=1,\ldots,l$

$$h_{i_k}^{n,1}(y) = \max(h_{i_{k-1}}^{n,1}(y), c_{i_{k-1}i_k}^l(\gamma_k)) \neq h_{i_k}^{n,1}(y_{i_k}^n(y))$$

$$h_{i_k}^{n,r}(y) = h_{i_{k-1}}^{n,r}(y) + c_{i_{k-1}i_k}^r(\gamma_k)) \neq h_{i_k}^{n,r}(y_{i_k}^n(y)),$$

$r=2,3$. Here

$$Z_p^n(y) = \{h_p^n(y_p^n(y))\} \cup \{(\max(h_v^{n,1}(y), c_{vp}^l(\alpha)),$$

$$h_v^{n,2}(y) + c_{vp}^2(\alpha), h_v^{n,3}(y) + c_{vp}^3(\alpha)) \mid (v,\alpha) \in Q_p^n(y)\}$$

$$Q_p^n(y) \subseteq Q_p(y) = \{(v,\alpha) | (v,p) \in E(y),$$
$$\alpha \in \Gamma_{vp}, c_{vp}^1(\alpha) \le y\}.$$

The value $y_p^n(y)$ and the sets $Q_p^n(y)$, $n=1,2,3$, $p \in V(y)$, are defined as:

a) for $n=2,3$ and $h_v^{n,1}(y^r(y)) \le y$,

$y_p^n(y) = y^r(y)$,

$\{(v,\alpha) \in Q_p(y) | c_{vp}^2(\alpha) < h_p^{n,2}(y^r(y)) - h_v^{n,2}(y),$

$c_{vp}^3(\alpha) > h_p^{n,3}(y^r(y)) - h_v^{n,3}(y)\} \subseteq Q_p^n(y)$;

b) for $n=3$ and $h_v^{3,1}(y^r(y)) > y$ or $n=1$

$y_p^n(y) = y^1(y)$,

$\{(v,\alpha) \in Q_p(y) | c_{vp}^2(\alpha) > h_p^{n,2}(y^1(y)) - h_v^{n,2}(y),$

$c_{vp}^3(\alpha) < h_p^{n,3}(y^1(y)) - h_v^{n,3}(y)$ or

$c_{vp}^1(\alpha) > y$ or $h_v^{n,1}(y) > y^1(y)\} \subseteq Q_p^n(y)$;

c) otherwise $y_p^n(y)$ is not defined and $Q_p^n(y) = Q_p(y)$.

It is easy to prove

Proposition 3. For all $y \in \{y_t\}$, $n=1,2,3$, and $p \in V(y)$

$$(y\ h_p^{n,2}(y) + h_p^{n,3}(y), h_p^{n,1}(y)) =$$
$$Lex\min\{(y \cdot f^2(x) + f^3(x), f^1(x) | x \in X, f^1(x) \le y\}.$$

For any vertex $p \in V(y)$, $n=1,2,3$, we can construct a parameterized route $x_p^n(y)$, which consists of d ($d>1$) parameterized paths

$x_{p_m}^n = x_{p_m}^n(H_p^n(y_{u_m})) = ((v_m = i_{m0}, \ldots, i_{ml_m}),$ $(\gamma_{m1}, \ldots, \gamma_{ml_m}))$, obtained for some $y_{u_m} \in \{y_t\}$ such that $v_1 = s$, $p_{u_d} = p$, $h_p^n(y_{u_d}) = h_p^n(y)$ and $u_{m-1} < u_m$, $p_{m-1} = v_m$, $h_{v_m}^n(y_{u_{m-1}}) = h_{v_m}^n(y_{u_m})$ for $m=2,\ldots,d$.

It is easy to prove that $x_p^n(y)$ is a parameterized path if in relationships (1) we accept $h_p^n(y) = h_p^n(y_p^n(y))$ iff $Z_p^n(y)$ does not contain a vector $(\lambda^1, \lambda^2, \lambda^3)$ such that

$$(y \cdot h_p^{n,2}(y_p^n(y)) + h_p^{n,3}(y_p^n(y)), h_p^{n,1}(y_p^n(y))) =$$
$$(y \cdot \lambda^2 + \lambda^3, \lambda^1)$$

6. Conclusions

For solving the problem **C**(w), a twolevel procedure can be used, which is similar to the proposed one for the initial procedure **A**. Inclusion the problem **C**(w) into general procedure for solving the initial problem **A** allows us to obtain "good" upper bounds of function $F(y)$. Taking into account such role of the problem **C**(w), it is sufficient to obtain its approximate solution. Furthermore, we can solve it even for some subpath containing vertex t.

The proposed approach can be improved for particular applications. For instance, it is possible to reduce the sets Γ_{vp} during solving problems **B**'(y).

The proposed approach can also be extended for more wide class of such problems. For example, a problem when function $f^1(x)$ is defined both operation "max" and "min" can be reduced [3] to the considered problem.

Acknowledgment

This work has been supported by the INTAS Project 96-820.

References

1. Gallo, G., and Pallottino, S., Shortest path algorithms, *Annals of Operations Research* (1988), 13, 1-4, 3-79.

2. Marsingh, D., and Chruin, P., Shortest-path algorithms: taxonomy and annotation, *Networks* (1984), 14, 2, 275-329.

3. Guschinsky, N.N., and Levin, G.M., Decomposition methods for minimization of a class of complex recurrent-monotonic functionals over parameterized paths in digraph, (1988), 57p. (Preprint of the Institute of Engineering Cybernetics, N 45). (in Russian).

4. Guschinsky N.N., Levin G.M., and Tanaev V.S., Parametric decomposition of problems to minimize complex functions over a set of parametrized paths in digraphs, *Izvestija AN SSSR, Technicheskaja kibernetika*, (1990), 6, 125-136 (in Russian).

5. Guschinsky, N.N., and Levin, G.M., Minimization of monotone superposition of recurrent-monotone functions on a set parameterized path in digraph, *Modeling Systems* (1991), 17, 167-178 (in Russian).

MULTI-SCALE IMAGE ANALYSIS and APPLICATIONS in FRAMEWORK of the INTAS 96-785 PROJECT

H.J.A.M. Heijmans
CWI, Amsterdam, The Netherlands
V.S. Kirichuk
Institute of Automation and Electrometry, Novosibirsk, Russia
V.A. Kovalev
Institute of Engineering Cybernetics, Minsk, Belarus
E.J. Pauwels
K.U.Leuven, Leuven, Belgium
M. Petrou
University of Surrey, UK
V. I. Tsurkov
Computing Center, Moscow, Russia
A.V. Tuzikov
Research-Engineering Center of Informational Technologies, Minsk, Belarus

The necessity of multi-scale methods for image analysis has been recognised since the emergence of the field of image processing. Processing images at different scales has many benefits as it gives additional information about the essence of the percept which we want to understand. Today, there exists various alternative approaches: linear and nonlinear image scale-spaces, morphological imaging techniques, quadtrees, pyramids, wavelets, fractal-based techniques, etc. The goal of the current project is to contribute to this important research area, and more ambitiously, to explore their interrelationships. Furthermore, we want to apply such methods to various concrete problems, in particular the analysis of medical images (texture analysis, segmentation).

An important application area where multi-scale methods can play an important role is Content-Based Image Retrieval (CBIR). Given the exponential growth of image and multi-media material in large databases and the Internet, there is increasing interest in search-engines that are able to retrieve images on the basis of image-oriented queries and visual clues. (Queries like: "Find images of deserted tropical beach" or "Find images that look like this one"). As compared to more traditionally oriented databases (where one looks for an exact match with the query-item), image databases are searched for objects similar to the query-item. Research has shown that more advanced techniques for automatic image analysis and more sophisticated ways of shape recognition will be necessary to perform such tasks.

Research activities

University of Surrey, UK
Institute of Engineering Cybernetics, Minsk, Belarus

1. Wavelets and 3D texture analysis.

Two approaches to the characterisation of 3D textures were studied. One based on gradient vectors and one on generalised co-occurrence matrices. They are investigated with the help of simulated data for their behaviour in the presence of noise and for various values of the parameters they depend on. They are also applied to several medical volume images characterised by the presence of micro-textures and their potential as diagnostic tools and tools for quantifying and monitoring the progress of various pathologies is discussed. The gradient based method appears to be more appropriate for the characterisation of micro-textures. It also shows

more consistent behaviour as a descriptor of pathologies than the generalised co-occurrence matrix approach. Results are presented in [1].

2. Content-based image retrieval.

Multimedia documents are different from traditional text documents because they may contain encoding of raw sensorial data. This fact has severe consequences for the efficient indexing and retrieval of information from large unstructured collections (e.g. WWW) since it is very difficult to automatically identify generic meanings from visual or audible objects. Two sub-problems of content-based image retrieval were studied at this stage, colour image retrieval and shape retrieval. Generalised co-occurrence matrices were employed in both cases.

Colour image retrieval. Our method of colour image retrieval is based on so-called colour co-occurrence descriptors that utilise compact representations of essential information of the visual content [2]. The set of descriptor elements represents "elementary" colour segments, their borders, and their mutual spatial distribution on the image frame. Such representation is flexible enough to describe image scenes ranging from simple combinations of colour segments to high frequency colour texture equally well. At the retrieval stage the comparison between a given query descriptor And the database descriptors is performed by a similarity measure. Image descriptors are robust versus affine transforms and several other image distortions. Basic properties of the method are demonstrated experimentally on an image database containing about 10,000 colour images.

Shape retrieval. We have suggested a method that exploits the orientation tokens that characterise a shape or an image for object identification [3,4]. The method is appropriate for identifying objects from their silhouettes or grey level texture. The approach was tested with the help of a large database of marine animals (mostly fishes). Pictorial queries by shape were answered successfully using only the L2 norm for histogram comparison. The accuracy of the answer can be improved if more features are computed from the histogram. Some example features for this purpose were proposed. On the other hand, the method was successfully tested for the identification of textures in leaves, using as feature a measure of the "order" that appears to be present in a texture. It was proposed that this can be expressed by the ratio of the number of orientation tokens that are more or less parallel to each other over the number of tokens that form larger angles with respect to each other. The pairs of tokens considered are at fixed distance from each other. For a quick search of an image database this produces an efficient method of sorting images "at a glance". For a more detailed description, one could use many different ranges of relative distance and thus produce a 2D angle-distance co-occurrence matrix for the description of the object. The use of histograms for such a task is ideal as histograms are rotation and translation invariant. On the top, they are scale invariant as well, because we consider all possible areas of tokens and bin their distances in a fixed number of bins, independent of the actual range of distances contained in each bin. Finally the use of orientation tokens is independent of any illumination and grey level changes, something of major importance when scanning a collection of images that might have been captured under varied conditions. As a corollary of this investigation, we have shown some evidence that the MRI images of healthy subjects show more organised textures than the MRI images of patients suffering from various brain disorders. Results are published in [1,4].

3. Non-rigid body registration of 2D/3D images.

A novel approach for 2D/3D image registration based on non-linear elastic deformations and global optimisation is proposed. Suitable cost function containing similarity, deformity, and discrepancy terms is suggested and experimentally studied. The terms of cost function may be used as measurements of the degree of deformation, necessary to bring two images in registration, and also as quantifiers of the evolution of various changes like as pathology development in medical images. Results are published in [5,6].

Computing Center, Moscow, Russia

The analytical model of the edge in the image improvement problem by means of anisotropic diffusion is considered. In 1990 parabolic type differential equations in partial derivatives were proposed by P. Perona and J.Malik [7] for two-dimension image processing. The main idea of this method is the following. As is well known, in the case of a problem with initial conditions for diffusion (thermoconductivity) equation the tendency of noise smoothing appears due to some specific properties of parabolic equations. Thus noise can be reduced in image processing. However, according to the same property of parabolic equations the image edges with infinite initial gradients would be also smoothed. This would lead to the blurring of images. In order to remove this drawback, P.Perona and J. Malik proposed to use a diffusion coefficient, which depends upon the gradient. It is important that the diffusion coefficient should tend to zero as the derivatives become infinite. In other words,

image noise should be eliminated while preserving infinite gradients, i.e. image edges. The Perona-Malik problem has put forward a number of publications. Among them regularisation is used in F. Cotte, P.-L. Lions, J.-M. Morel and T. Coll [8] and L. Alwarez, P.-L. Lions and J.-M. Morel [9]. The partial derivatives are represented as convolutions with Gaussians, which contain a small parameter. With this parameter tending to zero, the convolution tends to partial derivative. All mathematical transforms and numerical calculations are performed with the small parameter fixed (e. g. constant) and the solution is regular. In [9] it was stated that the limit solution is Lipschitz-continual for continual initial conditions as the small parameter tends to zero. Suggestion is made [9] that a divergent form of the Perona-Malik model may result in Adamar instability. That is why undivergent form of parabolic equations is proposed [9].

At this work the following results were obtained. Semi-similar solutions of anisotropic diffusion equations are found in one-dimension problem when a diffusion coefficient is a power function of the derivative. These solutions correspond to preserving of the infinite gradient with time. They are referred to as analytical edge model of anisotropic diffusion. It is found that the edge is preserved if the derivative power in the diffusion coefficient is more than 2. Otherwise, regular solutions take place. It is shown that the initial condition problem of the divergent Perona-Malik model is incorrect if the power degree number is more then two. We obtain that edge is a singular function of Gelder-continual class on the contrary to the Lipschitz-continual type limit solutions from [9]. An existence theorem for the weak (generalised) solution of the problem with initial conditions from Gelder-continual class is proved. Our prove is based on obtaining internal a-priori estimates for the first derivatives (or Lipschitz constants) of so-called potential functions, preserving regularity on edges. The weak solutions coincide with strong ones at points in which the diffusion coefficient is not equal to zero. At the edges the solution existence is considered as fulfilling of integral relationships with probe functions. A numerical experiment was conducted in which the initial edge considered as a Gelder-continues class function was perturbed by different types of noise. Results are the following. The edge is preserved and the image noise is reduced, thus Perona-Malik problem is solved. Semi-similar solutions of the anisotropic diffusion are obtained and analysed. They depend upon the ratio of a space co-ordinate to the square root of a time co-ordinate. Theorems on the asymptotic tending of the initial condition problem solutions to the solutions mentioned above are formulated in various topologies. Filter development on the base of the analytical edge model needs utilisation of discrete step numerical algorithm. Implicit discrete schemes of parabolic type equations are used in one-dimension calculations. Edge localising is performed with an interpolation procedure.

Research-Engineering Center of Informational Technologies, Minsk, Belarus

Research directions:

- Granulometries, pattern spectrum, and the opening transform;
- Morphological image analysis (shape similarity and symmetry measures, fast morphological algorithms, connected operators, texture classification);
- Content-based image retrieval.

Obtained results.

1. Similarity and symmetries of arbitrary polygonal shapes were investigated [13,14,15]. Notion of symmetry measure was introduced. Contour representation known as a turning function is used to evaluate symmetry measure. This representation is suitable for simply connected compact objects. Fast algorithms for similarity and symmetry measures can be implemented using such a representation. Analysing such functions we can compute a measure of reflection symmetry (with position of the best symmetry axis) or a measure of rotational measure (for different orders of rotations simultaneously). For affine symmetry the main idea consists in using canonical form normalisation in such a way that affine (skew or skew rotation) symmetry of original shape is equivalent to usual (reflection or rotation, respectively) symmetry of its canonical form. Algorithm of finding similarity and symmetry measure using turning function was implemented in C++ and tested for different objects and transformations (reflections, rotations, or affine transformations). The influence of noise to symmetry measures was tested also. User interface for this software was implemented in Java.

2. Symmetries of 3D convex shapes were investigates [16]. Symmetry measures based on Minkowski and Brunn-Minkowski inequalities for volumes and mixed volumes of convex objects were introduced for evaluations of symmetry degree. For the case of convex polyhedra functionals used in these measures have a nice property of piecewise concavity which allows to reduce the complexity of optimisation problems. For reflection symmetry of polyhedra the problem is reduced to checking of the functional value for the critical rotations of polyhedra, number of which seems to be finite. For rotation symmetry of order 2 the problem is the

same as for reflection symmetry. For rotation symmetry of orders more than 2 no optimisation method is proposed. The case of mirror rotation symmetry is equivalent to finding similarity measure of two polyhedra.

Implementation of polyhedra similarities and symmetries is now in progress. Several program components including finding of critical rotations for fixed axis are already written and tested using Matlab system.

Institute of Automation and Electrometry, Novosibirsk, Russia

Relief reconstructing and mobile objects detecting by sequences of images.

The data of continuous synchronous observation of analysed scene from several satellites are intended to be used with the purpose of detecting of dynamic processes occurring on the Earth surface and at the adjacent atmosphere layers (volcano activity, tornado, typhoons, explosions, conflagrations, etc.). The observing simultaneously and from different points provides selectivity of the atmospheric phenomena in the depth direction; the observing continuously enables to reveal their dynamic features.

Registration system geometry is not definite completely in the common case. So far, the problem of estimation of cameras orientation should be solved to restore the structure of the scene observed. Most approaches are based on relations between co-ordinates of corresponding points of projected images. Standard estimating fundamental matrix methods are not efficient because of cameras with small angles of view are used to get high spatial resolution. A great number of images in sequence observed and the knowledge about mathematical model that describes dynamics of the system geometry gives us new opportunities in sequences combined processing but requires new methods of data processing.

Two cases are investigated within the framework of the problem: a) the registration system is fixed in respect to a scene (geostationary satellites); b) the registration system is mobile (orbital systems).

The main purpose is to detect extremely mobile, slightly contrast objects when analysing data being received from geostationary system. Its complexity is conditioned by an image complexity -- an image contains stationary background (response from the Earth surface), quasistationary part (response from clouds), response from an objects and a noise. Moreover the intensity of response from an objects is much lower then both the intensity of response both from clouds and from the Earth surface. Besides there exist little shifts of registration system. Consequently every preventive factor should be reduced extremely to achieve good results in detecting an objects and their spatial location.

Thus, to solve the general problem one should solve several separate ones:

- to bind the sequence to some fixed system of co-ordinates using a correlative criterion;
- to carry out interframe processing to reduce powerful responses from the Earth surface and from clouds. The algorithm of interframe processing is based on monosemantic expanding of every current image with a set of binary images (basis functions), approximating an every consequent image with the functions and, finally, getting a corresponding differential image that does not contain stationary the background;
- to select (by means of linear and non-linear filtering) the points probably belonging to the object;
- to calculate an area of spatial location for every point detected for every sequence;
- finally, to combine results of processing for all sequences analysed and to select the points that areas of spatial location are intersected. The points are supposed to belong to the object(s).

When a registration system is mobile it is necessary to estimate parameters of the system (as a rule, cameras orientation is indefinite) under condition of extremely small angles of view (~1°), and to find corresponding points under conditions of great difference in angles of observation (up to 90°) and of great distance from the scene observed (~1000 km).

There are algorithms listed below designed to solve the problem:

- of finding corresponding points on the base of modified, geometrically tuneable to projective deformations, correlative and morphological criteria.
- of estimating geometry both of cameras location and of scene on the base of combined testing sequence images and solving essentially overdetermined system of algebraic equations.
- of restoring relief by sequences of images on the base of sets of corresponding points using LMS method.
- of global minimising the functional with the definite geometry of sequence and without finding of corresponding points.

Obtained results:

1. The algorithms of combined processing of sequences obtained from geostationary satellites are designed aimed to detecting extremely mobile, slightly contrast objects. It is efficient under conditions of time instability and powerful background signal (false alarm probability is $5*10^{-5}$, detection probability is 0.95).

2. The algorithm of restoring relief from a sequences of images under conditions of indefinite cameras observation angles is designed. Its accuracy is examined on models and tests. It achieves 65 meters when reconstructing relief of Erebus volcano by 24 images obtained from the distance of 1100 ÷ 1500 km (relief height ~0÷4 km) .

3. The algorithm of surface reconstruction by long (~100 images) sequence of images and difference of cameras observation angles 30° ÷ 40° is developed. Its accuracy achieves 1/n where «n» is number of images in the sequence. The algorithm uses a procedure of global minimising of the selected functional instead of extremely unstable procedure of corresponding points finding.

CWI, Amsterdam, The Netherlands

There exists increasing interest in the development of tools that consider images at different scales of resolution. In this respect pyramidal image structures are of particular interest. A well-known instance of such a structure is the wavelet transform, but there exist others, such as the Gaussian pyramid. In collaboration with Prof. J. Goutsias (Johns Hopkins University, Baltimore), Heijmans is investigating pyramid structures based on morphological operators [17,18]. One of the ambitious goals of this project is to make a systematic study of nonlinear (morphological) wavelets.

Connected operators form a special class of morphological operators which act on the level of flat zones of images rather than on individual pixels. As a consequence, such operators can delete edges, but cannot change them, neither their shape nor their location. As a result, connected operators are well-suited for many imaging tasks, such as segmentation, filtering, and coding. The aim of this project is to develop a consistent mathematical theory for connected operators, and to investigate their implementation using a tailor-made graph structure. In 1998, the activities in this project have been very limited, but they will be taken up again in 1999.

Scale-spaces are a modern bottom-up tool in computer vision. In scale-space theory one embeds an image into a continuous family of gradually smoother versions of it. Increasing the scale should then simplify the image without creating spurious detail. The prototype example is the Gaussian scale-space, the construction of which is based on filtering the image with Gaussians of increasing width. Alternatively, one can view the resulting family of images as a solution of a linear diffusion equation. Recently, various nonlinear scale-spaces have been constructed, including those based on morphological operators. Together with van den Boomgaard of the University of Amsterdam, Heijmans has proposed a new axiomatic framework for scale-spaces based on algebraic concepts. This framework includes various existing examples, including Gaussian and morphological scale-spaces. A technical report is in preparation.

Katholieke Universiteit Leuven, Belgium

Within the framework of this project Content-Based Image Retrieval (CBIR) has been identified as an application that serves as a testbed for the theoretical elaborations of the multi-scale paradigm in image analysis. We recall that CBIR deals with the problem of automatically retrieving images that are perceptually similar to a given query-image from a large digital databank.

As a starting point we therefore looked at number of existing systems, such as QBIC (IBM), Virage, Photobook (MIT), etc,... to clearly identify the current state-of-the-art. We found that most of these systems are guided by pixel-based similarity measures. However, extensive experimentation over the last few years has shown that matching natural images solely on that basis of global similarities is often too crude an approach to produce satisfactory results. What is required is some form of perceptually relevant segmentation that allows one to identify a (small) number of salient and semantically meaningful image-regions which can then serve as the basis for more discerning region-based matching, e.g. it stands to reason that when it comes to perceptual matching, the foreground in an image is far more important than the background.

We therefore concentrated in this first part of the project on a generic segmentation-methodology that, using a number of basic features such as colour and texture, divides an image in a small number of perceptually salient regions, which can then be used as the starting point for similarity matching. In essence the methodology is very simple: after mapping the pixels of an image into a feature space (e.g. colour-space), we can proceed to cluster the data in this feature space. Identifying the thus-obtained clusters in the original image, yields a segmentation.

Hence, from an abstract point of view, segmentation and perceptual organisation can be interpreted as a problem of selecting appropriate features, followed by cluster-detection in feature-space. In fact, we can tighten up this argument even further since both steps are but two aspects of the same problem, as a particular feature-space is deemed appropriate whenever it shows pronounced clusters. Indeed, if mapping the pixels into the feature-space lumps them all together, this particular set of features is obviously of little use in defining perceptual saliency.

However, clustering problems commonly encountered in CBIR-applications are particularly challenging as mapping the images to feature-spaces often produces very irregular and unbalanced data-clouds. Furthermore, given the fact that segmentation and region-extraction should proceed automatically, we cannot assume that prior knowledge about the number of clusters or their shape is available. We have therefore developed a robust and versatile nonparametric clustering algorithm that is able to handle the unbalanced and highly irregular clusters encountered in such CBIR-applications. The strength of our approach lies not so much in the clustering itself, which is based on non-parametric density-estimation, but rather in the definition and use of two cluster-validity indices that are independent of the cluster-topology. By combining them, an optimal clustering can be identified, and experiments confirm that the associated clusters do indeed correspond to perceptually salient image-regions [19].

For more details and examples, we refer to our website http://www.esat.kuleuven.ac.be/~frederix/segmen-tation.html .

In the second part of our work we focussed on the problem of classification. The relevance for CBIR is clear: once prototype-images have been put into different groups, new images need to be classified into the existing classes so that they can be indexed and identified. There is a huge literature on classification and Support Vector Machines (SVM), one of the more recent methodologies, has enjoyed a lot of attention in recent years. It uses the training data to construct an optimally separating hypersurface between two classes. This surface is represented by a small number of "support vectors" that are used to classify new data. If the data are linearly separable, these support vectors can be determined directly. If this is not the case, SVM sidesteps the non-linearity by mapping the data into a higher dimensional space where a linear separator can be found. However, the nature of this higher dimensional space is highly data-dependent and finding an appropriate transformation is often difficult and left to the user. We have developed a new method that finds support-like vectors (for which we coined the name "sentinels") in the original data-space (so there is no need to find an appropriate transformation to a higher dimensional space). It is similar to LVQ and k-means in that it is based on attraction between data-points. However, unlike these aforementioned methods, it identifies datapoints near the boundaries of the classes, which can then be used for classification. A paper describing this algorithm and its performance is in preparation.

References

1. V.A. Kovalev, M. Petrou and Y.S. Bondar. Using orientation tokens for object recognition, Pattern Recognition Letters, Vol. 19, No. 12, pp. 125-132, 1998.
2. V. Kovalev and S. Volmer. Color Co-occurrence descriptors for querying-by-example, The International Conf. on Multimedia Modeling (MMM'98), Oct. 12-15, 1998, Lausanne, Switzerland, IEEE Comp. Society Press, pp. 32-38, 1998.
3. V.A. Kovalev, M. Petrou and Y.S. Bondar. Texture anisotropy in 3D images, IEEE Trans. on Image Processing, Vol. 8, No. 3, pp. 346-360, 1999.
4. V.A. Kovalev and M. Petrou. Image retrieval by object shape using co-occurrence matrices, 5th Int. Conf. On Pattern Recognition and Information Processing (PRIP'99), Minsk, Belarus, May, 1999 (in Russian, in press).
5. V.A. Kovalev, A method of elastic exponential deformations for image registration,(in Russian, in press).
6. V.A. Kovalev, Registration of 2D and 3D images of non-rigid objects, (in Russian, in press).
7. P. Perona and J. Malik Scale-space and edge-detection using anisotropic diffusion. IEEE Trans. on Pattern. Anal. and Mach. Intel. Vol. 12, N 7, pp. 629-639, 1990.
8. F. Cotte, P.-L Lions, J.-M. Morel and T. Coll. Image selective smoothing and edge detection by nonlinear diffusion I.SIAM J. Numer. Anal., Vol. 29, N1, pp. 182-193, 1992.
9. L. Alwarez, P.-L Lions and J.-M. Morel. Image selective smoothing and edge detection by nonlinear diffusion II. SIAM J. Numer. Anal., Vol. 29, N 3, pp. 845-866, 1992.

10. V.S. Kirichuk, V.P. Kosykh, G.I. Peretjagin. Extrinsic and Intrinsic Parameters Estimation for Stereo-Vision System with a Common Observation Point (Long-Focal Approximation). 4th All-Russian with invited foreign participants Conf. on Pattern Recognition and Image Analysis (ROAI-98), Akademgorodok, Novosibirsk, Russia, Oct 11-18, 1998 (in Russian).
11. V.P. Kosykh, G.I. Peretjagin. Recovering the 3-D Structure of Surfaces on Sequence of the Images Obtained from a Moving Camera Directed to the Same Point of Scene. 4th All-Russian with invited foreign participants Conf. on Pattern Recognition and Image Analysis (ROAI-98), Akademgorodok, Novosibirsk, Russia, Oct 11-18, 1998 (in Russian).
12. V.S. Kirichuk, V.P. Kosykh, G.I. Peretjagin. Scene structure and camera motion estimation using long image sequences (longfocal approach). Pattern Recognition and Image Analysis (accepted).
13. S. Sheynin, A. Tuzikov, D. Volgin. Computation of Symmetry Measures of Polygonal Shapes. Proceedings of CAIP'99, Ljubljana, Lecture Notes in Computer Science, Springer, 1999.
14. S. Sheynin, A.Tuzikov, and D. Volgin. Symmetry measures of polygons based on the turning function. (Russian) – Pattern Recognition and Information Processing, Minsk, 1999, Minsk-Szczecin, 1999, Vol.2, pp.90-94.
15. S. Sheynin, A.Tuzikov, and D. Volgin. Symmetry measures computation for polygonal shapes. Pattern Recognition and Image Analysis, N.3, Vol.9, 1999.
16. A. Tuzikov and S. Sheynin. Symmetry measures for 3D convex shapes. In: D. Chetverikov, T. Sziranyi (eds). Fundamental Structural Properties in Image and Pattern Analysis 1999, International workshop, Budapest, Österreichische Computer Gesellschaft, 1999, band 130, pp.77-86.
17. J. Goutsias and H. J. A. M. Heijmans. Nonlinear multiresolution signal decomposition schemes. Part 1: Morphological Pyramids. IEEE Transactions on Image Processing (to appear).
18. H. J. A. M. Heijmans and J. Goutsias: Multiresolution signal decomposition schemes. Part 2: Morphological Wavelets. CWI Research Report PNA-R9905, Amsterdam 1999.
19. E.J. Pauwels and G. Frederix: Finding salient regions in images: Non-parametric clustering for image segmentation. Computer Vision and Image Understanding. Special Issue on Content-Based Access of Image and Video Libraries, June 1999, 16 pages.

Table of Contents

1. Theory

2. Robotics and Intelligent Control

3. Signal Processing

4. General Application

5. Pattern Recognition and Image Processing

6. INTAS

Научное издание

International Conference on Neural Networks and Artificial Intelligence

PROCEEDINGS

12-15 October 1999

Международная конференция по нейронным сетям и искусственному интеллекту

Труды научно-технической конференции ICNNAI'99 профессорско-преподавательского состава, аспирантов и студентов (12-15 октября 1999 года)

на английском языке

ISBN 985-6584-05-1

Редактор Т.В. Строкач
Художественный редактор А.П. Дунец
Компьютерный дизайн и верстка А.П. Дунец

Подписано в печать 26.09.99. Формат 60x84/8. Бумага UniPaper. Гарнитура Times New Roman. Усл. печ. л.26,0 Уч.-изд. л.28,0 Заказ № 584 Тираж 100 экз. Отпечатано на ризографе Брестского политехнического института. 224017, г. Брест, Московская, 267.