Izabela REJER[1], Krzysztof LORENZ[2]

West Pomeranian University of Technology, Szczecin (1), West Pomeranian University of Technology, Szczecin (2)

doi:10.15199/48.2015.02.24

Classic genetic algorithm vs. genetic algorithm with aggressive mutation for feature selection for a brain-computer interface

Abstract. *The classic genetic algorithm has been successfully applied to many optimization problems. However, its usefulness is limited when it comes to feature selection, particularly if a high reduction rate is expected. The algorithm, in its classic version, returns feature sets containing approximately 50% of the total number of features. In order to decrease this rate, a penalty term penalizing individuals of too many features is often added to the fitness function. This solution seems to be reasonable but, as will be shown in this paper, provides only a slight improvement in the reduction rate. In order to obtain a satisfactory classification accuracy and a high reduction rate, not only the fitness function but also other algorithm elements must be reconsidered.*

Streszczenie. *Klasyczny algorytm genetyczny był z powodzeniem stosowany w wielu problemach optymalizacyjnych, jednakże jego użyteczność jest ograniczona w problemach selekcji cech, zwłaszcza jeżeli wymagana jest wysoka stopa redukcji cech. Algorytm, w jego klasycznej wersji, zwraca zbiory cech zawierające około 50% pierwotnej liczby cech. W celu zmniejszenia tej liczby, do funkcji przystosowania algorytmu dołącza się często człon kary, karzący osobniki kodujące zbiory o zbyt dużej liczbie cech. Takie rozwiązanie wydaje się być rozsądne, ale, jak zostanie to przedstawione w artykule pozwala jedynie na niewielką poprawę stopy redukcji. Stąd, w celu uzyskania satysfakcjonującej dokładności klasyfikacji i wysokiej stopy redukcji, nie tylko funkcja przystosowania, ale również inne elementy algorytmu muszą zostać wzięte pod uwagę (**Klasyczny algorytm genetyczny, a algorytm z agresywną mutacją w procesie selekcji cech na potrzeby interfejsu mózg-komputer)**.*

Keywords: brain-computer interface, BCI, genetic algorithm, feature selection, motor imagery.
Słowa kluczowe: interfejs mózg-komputer, BCI, algorytm genetyczny, selekcja cech, wyobrażenie ruchu.

1. Introduction

According to the strict definition the brain-computer interface (BCI) is a system in which a user sends the control commands or text messages with the exclusive use of the brain activity. There is also a more liberal approach according to which two types of BCIs can be distinguished, a dependent BCI and an independent BCI. While in the dependent BCI the brain activity is recorded together with other types of activity, such as eye lids or face muscles movements, the independent BCI allow only the brain waves to be recorded. Regardless of the BCI type, in order to decide which command has been intended by the user, the brain activity has to be measured, usually with EEG device, and analyzed.

The automatic analysis of EEG signal is very difficult for two main reasons. First, the signal is almost always contaminated with artifacts, second, there is a huge space of possible features describing this signal. In order to deal with the first problem methods for removing artifacts are used (e.g. spatial filters, frequency filters, Independent Component Analysis etc. [1, 2, 3]). To address the second problem the methods for feature selection have to be applied (e.g. step-wise selection [4, 5], ReliefF [6], Correlation-based Feature Selection [7], genetic algorithms [8], LASSO [9], random selection [10] etc.). The paper regards the second problem, i.e. the feature selection problem.

There are at least three main reasons why feature selection is necessary when decoding EEG signals for BCI purpose. The first of them is a very limited number of training examples that can be gathered for the classifier training. Since the brain waves are subject specific, it is impossible to gather data from a set of subjects and to train a classifier on the joined data set. Just on the contrary, each subject must obtain his own classifier, adapted to his specific brain activity patterns. As a result of a habituation phenomenon, changes in the subject's mental condition, health problems etc., the number of trials that can be performed with one subject is very limited. Usually only about 100-300 observations are recorded from one subject. Comparing this number to the thousands of features that can be extracted from raw EEG signal, it is obvious that for a proper classifier training a huge reduction of the feature space is necessary.

The second reason for using feature selection techniques when designing a BCI system is the possibility of decreasing the number of electrodes applied on the user head. If during the feature selection process all the features extracted from some channels were discarded as unimportant for the classification precision for the given subject, these channels could be eliminated from the BCI system prepared for this subject. Since fewer channels (and so fewer electrodes) means less time for the interface application, the massive feature elimination can increase the user comfort and decrease the overall cost of the interface.

On top of that, due to elimination of unnecessary features, the interface can respond faster to the user mental acts. This is not only because fewer features needs less time to be extracted but most of all because of the possibility of eliminating features describing phenomena that need time to evolve. A good example here is an event relation synchronization (ERS) in β frequency band that can be detected in EEG signal about 5-6 seconds after the movement onset. If all the features describing β ERS phenomenon were eliminated from the feature space during feature selection, the time needed for the analysis of each portion of EEG signal recorded from the given subject could be shorten of about two-three seconds.

There are a lot of methods that can be used for feature selection. Some examples were given at the beginning of the paper. One of the method that is popular in BCI research is that of genetic algorithms (GAs) [8, 11, 12, 13]. Theoretically, different genetic algorithms can be used in the feature selection process (e.g. algorithm coding all features [8, 11], Culling algorithm [12, 13], nondominated sorting genetic algorithm-II (NSGA-II) [14], algorithm with a limited number of features [15], etc.). In practice however, usually the simplest approach, i.e. the algorithm that codes all the features within each individual guided by pure classification accuracy, is used. This approach is easy to implement, but has one weakness: by using pure classification accuracy as a fitness function, the optimization algorithm tries to find individuals with 100%

classification accuracy, regardless of their number of features. Hence, when a genetic algorithm is used in the feature selection process, more sophisticated approaches to coding individuals or more sophisticated fitness functions should be used.

One of the possible solutions is to extend the fitness function by using a penalty term, penalizing individuals coding too many features. This is usually a good solution but the question is whether it will bring expected results when a very small feature set will be searched. The aim of this paper is just to answer this question. In order to deal with this task the genetic algorithm with extended fitness function will be used to select features from a set of 324 features extracted from a data set submitted to the second BCI Competition (data set III – motor imagery) by Department of Medical Informatics, Institute for Biomedical Engineering, Graz University of Technology [16]. The results returned by this algorithm will be compared with results obtained with two genetic algorithms guided by pure classification accuracy, the classic genetic algorithm and the genetic algorithm with aggressive mutation.

2. Methods

Three genetic algorithms were used in the survey: the classical genetic algorithm proposed by Holland, the classic genetic algorithm with a penalty term in the fitness function, and the genetic algorithm with aggressive mutation.

2.1. Classic genetic algorithm

The classical genetic algorithm is one of the most popular genetic algorithms used for feature selection. In this approach, an individual contains only one chromosome of the length equal to the number of all the extracted features. Each gene holds information about whether the corresponding feature is present in the solution that is encoded within the individual (allele 1) or not (allele 0). For example if the feature space contains 10 features, an individual of a genotype 0110001101 encodes the set of features composed of features: 2, 3, 7, 8, and 10. The scheme of the classical genetic algorithm is as follows. First the initial population of individuals is randomly drawn from the whole set of possible solutions. Next the individuals are evaluated according to the chosen fitness function and the stop condition of the algorithm is tested. If the stop condition is not met, the selection process is started. There are a lot of different selection approaches but in most of them, the individuals of the higher value of the fitness function have higher chances to be chosen to the mother population. The individuals from the mother population are then processed according to the applied genetic operators. Usually two genetic operations are performed one after another, crossover and mutation. After the second operation, the final population composed of the individuals born during genetic operations is created and the whole cycle of evaluation, selection and genetic transformations is repeated until the stop condition of the algorithm is met.

2.2. Classic genetic algorithm with a penalty term

When the classic genetic algorithm is used for feature selection, the individuals are evaluated according to the classification accuracy that can be obtained with features encoded in these individuals. That means that for each individual a classifier is created. This classifier takes as inputs all the features encoded in the individual. After the classifier training, its accuracy is evaluated and used directly as a fitness value of the individual. The problem is that the feature selection process is not aimed at finding a feature set of the highest classification accuracy, but at finding the smallest possible feature set of the sufficiently high classification accuracy. Considering the real goal of the feature selection process, the fitness function based only on the classification accuracy is not the best choice.

The approach that can be taken to make the classic genetic algorithm more suited to the real goal of the feature selection process is to extend the fitness function of the genetic algorithm by adding an additional component, penalizing the individuals for coding too many features. Assuming that both terms (the accuracy term and the penalty term) have the same weight, the fitness function of the genetic algorithm can be given as follows:

$$F = 0.5A + 0.5\frac{T-O}{T} \tag{1}$$

where: *F* – fitness function, *A* – classification accuracy, *T* – number of all the extracted features, *O* – number of features encoded in the individual.

The additional term in the fitness function is the only difference between the algorithm from the previous section and the algorithm described here. All the algorithm steps from the random choice of the initial population to the applied genetic operators remain the same.

2.3. Genetic Algorithm with aggressive mutation

After adding the penalty term to the fitness function the optimization process has two goals: minimization of the value of the penalty term and maximization of the classification accuracy. Simultaneous two-direction search is difficult and needs much more time for finding the satisfactory result than one-direction search. The question is whether this two-direction search is really necessary. Often when the feature selection process is performed, the number of features that can be applied in the classifier (because of the technical background, the user's needs or other reasons) is known in advance. In such situation the goal of the selection process changes once again and now is as follows: for the given number of features, find the feature set that provides the satisfactory accuracy. Such a situation is well known in the BCI domain where because of a very small number of observations that can be gathered for the classifier training, only a few features are often allowed.

A genetic algorithm that can be used when the number of features is given in advance is the algorithm with aggressive mutation (GAAM), proposed a year ago by one of the authors of this paper [17]. This algorithm uses the integer coding system, where one gene contains an index of one feature from the feature set encoded in the individual. Hence, the number of individual's genes is equal to the number of features required by the user. The integer coding scheme is the first feature that differs GAAM from the classic GA. The next is the order of two the most important steps of the genetic algorithm, selection and reproduction. In GAAM the reproduction takes place before the selection. During the reproduction step, the mother population is enlarged by adding new born individuals and during the selection step the population is reduced to its original size by selecting individuals with the highest value of the fitness function.

The last difference between GAAM and the classic genetic algorithm is the mutation operation. In GAAM the mutation scheme is very aggressive - not only each individual in the population is mutated, but also each of gene of that individual. To be exact the mutation scheme is as follows:

```
for $i$=1 to $M$
    take an individual $i$
    for $g$=1 to $N$
```

```
        take a gene $g$
        assign a random value from the interval {0,1,2...$P$}
        to the gene $g$
        save the individual $i$ as a new individual.
```

where: M - number of individuals in the mother population, N - number of genes in an individual, P - total number of the extracted features.

3. Experiment settings

The benchmark data set used for the algorithms' comparison [16] was recorded from a normal subject (female, 25y) whose task was to control the movements of a feedback bar displayed on a screen by means of imagery movements of the left and right hand. During the experiment EEG signals were recorded. They were measured over three bipolar EEG channels (C3, Cz and C4), sampled at 128Hz and preliminarily filtered between 0.5 and 30Hz. The whole data set contained 280 trials, but only 140 of them were published with target values (1 - left hand, 2 - right hand).

In order to obtain a feature set for the algorithms' comparison, 324 frequency band power features were extracted from the raw EEG signals. The band power was calculated individually for:

- 12 frequency bands: alpha band (8-13Hz), five sub-bands of alpha band (8-9Hz; 9-10Hz; 10-11Hz; 11-12Hz; 12-13Hz), beta band (13-30Hz), and five sub-bands of beta band (13-17Hz; 17-20Hz; 20-23Hz; 23-26Hz; 26-30Hz),
- each of 9 seconds of the trial,
- each of 3 channels (C3, Cz, C4).

The main aim of the survey reported in this paper was to reduce the whole set of 324 features to the set of only few features providing a high classification accuracy. In order to evaluate the classification accuracy of each of the feature sets processed during the survey, a classic linear SVM classifier was used [18, 19]. To adapt the data set to the SVM classifier requirements, the class labels were changed to -1 for the left hand and to 1 for the right hand. The classification threshold was set to 0, such that all the classifier results greater than 0 were classified as *right hand* and all the results smaller or equal to 0 were classified as *left hand*.

In the case of all three algorithms a separate classifier was built for each feature set from each population. Each classifier was trained over the training set containing 80% of data. The remaining 20% was reserved for the testing process. During the classifier training the 10-fold cross-validation scheme was used. To calculate the accuracy term for the fitness function the mean value calculated over the classification accuracy obtained for all 10 validation subsets was used.

Each time when the genetic algorithm stopped, the individual with the highest value of the fitness function was returned. The indexes of features encoded in this individual were then introduced as inputs to the final classifier that was trained over the whole training set and tested over the testing set. If the accuracy calculated over the training set was higher than the accuracy calculated over the testing set, the classifier was marked as an overfitted one and excluded from further comparisons. In the opposite case, the average classification accuracy over the whole set of 140 observations was calculated as the final measure of the classifier performance.

4. Results and discussion

At the first stage of the survey the classic genetic algorithm guided by pure classification accuracy was run. Since genetic algorithms generally find sub-optimal solutions, the algorithm was run ten times. The main parameters of this algorithm were set as follows: 100 individuals, 1 chromosome per individual, 324 genes per chromosome, 250 iterations. Each individual coded one subset of the feature set. The initial population was chosen randomly (using uniform probability distribution). Two classic genetic operations were used: one-point crossover (with a probability of 0.8) and one-gene mutation (with a probability of 0.025). Individuals that proceeded to the next population were selected according to the tournament method. Each tournament was run among the random number of individuals chosen randomly from the mother population. Table 1 presents the results of the best individuals from each run. The succeeding columns of the table denotes: No. of features: number of features encoded in the individual; Train, Test and Total: the classification accuracy of the final classifier calculated over the training, testing and the whole set, respectively. Total is equal to "-" for overfitted classifiers.

Table 1. Results obtained from ten runs of the genetic algorithm guided by pure classification accuracy

	No. of features	Train [%]	Test [%]	Total [%]
1	151	100	85.71	-
2	151	100	92.86	-
3	170	100	89.29	-
4	153	100	92.86	-
5	153	100	96.43	-
6	152	100	92.86	-
7	144	100	92.86	-
8	157	100	89.29	-
9	165	100	92.86	-
10	153	100	85.71	-

The second algorithm used in the survey was the genetic algorithm with a penalty term in the fitness function. The algorithm parameters, except from the fitness function were set at the same levels as in the first stage. The fitness function was composed of two terms, the accuracy term and the penalty term. Both terms have equal weights. The algorithm results returned in all ten runs are presented in Table 2. The succeeding columns of the table denotes: Fitness: value of the fitness function; Accuracy: classification accuracy calculated over 10 validation sets; Penalty: value of the penalty term of the fitness function; No. of features: number of features encoded in the individual; Train, Test and Total: the classification accuracy of the final classifier calculated over the training, testing and the whole set, respectively. Total is equal to "-" for overfitted classifiers.

At the last stage of the experiment the algorithm with aggressive mutation was applied to reduce the feature set. In order to decide how many features should be introduced to the individuals, Raudys recommendation [20] was taken into account. According to this recommendation in order to train a classifier properly at least 10 times more observations per class as the features should be gathered. In view of a data set composed of 140 observations, an individual should encode about 6 features. Since the coding scheme used in the algorithm with aggressive mutation is based directly on feature indexes, an individual that encodes 6 features must have 6 genes. The remaining parameters of the algorithm were: 30 individuals and 100 iterations. With these settings, the selection pool in each population was composed of 240 individuals (30 mother individuals, 180 individuals created during mutation, and 30 individuals created during crossover). The algorithm results are presented in Table 3. The succeeding columns of the table denotes: Indexes of features: indexes of features encoded in the individual; Train, Test and Total: the classification accuracy of the final classifier calculated over the training, testing and the whole set, respectively.

Table 2. Results obtained from ten runs of the genetic algorithm with a penalty term in a fitness function

	Fitness	Accuracy	Penalty	No. of features	Train [%]	Test [%]	Total [%]
1	74.53	88.57	60.49	128	100	82.14	-
2	75.71	90.00	61.42	125	100	85.71	-
3	76.23	88.57	63.89	117	100	75.00	-
4	75.00	88.57	61.42	125	100	85.71	-
5	74.53	88.57	60.49	128	100	82.14	-
6	75.51	89.29	61.73	124	100	85.71	-
7	75.19	91.43	58.95	133	100	89.29	-
8	74.74	87.14	62.35	122	100	92.86	-
9	75.30	88.57	62.04	123	100	89.29	-
10	75.01	84.29	65.29	111	100	78.57	-

Table 3. Results obtained from ten runs of the genetic algorithm with agressive mutation

	Indexes of features	Train [%]	Test [%]	Total [%]
1	5,24,26,96,104,156	94.83	100.00	95.86
2	5,24,34,96,104,321	94.83	96.43	95.15
3	24,86,104,133,156,181	93.10	96.43	93.77
4	6,24,86,104,181,196	92.24	100.00	93.79
5	5,24,96,104,209,321	93.97	96.43	94.46
6	5,24,57,96,104,321	93.10	96.43	93.77
7	5,24,26,96,104,156	94.83	100.00	95.86
8	5,23,24,107,133,156	92.24	96.43	93.08
9	5,26,58,104,199,216	90.52	100.00	92.42
10	5,24,34,96,104,321	94.83	96.43	95.15

According to the results presented in Tables 1 and 2 both classic genetic algorithms are not very useful when comes to find a feature set of a high classification precision and small number of features. Both GAs returned feature sets composed of much too large number of features. Of course the results returned by the algorithm with a penalty term were slightly better (this GA eliminated on average about 20% of features more than the classic GA), however, still the size of the feature set in the best case was equal to 111 features. Moreover all classifiers built over the feature sets returned by both algorithms were overfitted

The results returned by GAAM were very good in all ten runs. The average classification rate across ten runs was about 94%, and none of the classifiers were overfitted. Moreover, the feature sets returned in different runs were similar. Some of the features reappeared even in more than 50% of all the chosen feature sets.

The results of the GA with a penalty term were surprising for the authors of this paper and therefore a lot of additional experiments were performed. The goal of these experiments was to find out whether different weights assigned to both terms of the fitness function would allow to increase the impact of the penalty term on the optimization process. None of these experiments gave the satisfactory results. Regardless of the weight of the penalty term, the number of features in the individual was always higher than 100. That proves that introducing a penalty term to the fitness function is not enough to obtain a genetic algorithm that is able to reduce the feature space in a high degree. The problem is that the whole scheme of Holland GA is not suited for this task. Why? The first problem is with the initial population. If it is drawn randomly (using uniform probability distribution) about 50% of individuals' genes are equal to one, and the remaining genes are equal to zero. That means that each individual encodes about 50% of all the features from the feature space. Since the probability that a large number of ones will be discarded from the population with the classic crossover and the classic mutation is very low, the algorithm from the beginning prefers individuals of the higher classification accuracy.

The solution could be to start the algorithm from the individuals encoding only the given number of randomly chosen features. In such the case however, another problem would arise. With a such scheme it would be much easier for the algorithm to obtain better classification precision after increasing the number of features than after decreasing it. Of course this time the algorithm would stop with a smaller number of features than when using uniform probability distribution but this number would be much higher than that given at the beginning of the algorithm.

That best solution is when the algorithm starts with a small number of features and sticks to this number all the time. This, however, needs the changes at least in the coding approach and in the genetic operations. The algorithm that fulfills both requirements is the algorithm with aggressive mutation. This algorithm uses the coding scheme that allows to determine the number of features in the feature set and uses the mutation scheme that guarantees a high variability of the succeeding populations.

5. Conclusion

Among all three genetic approaches to feature selection applied in the reported survey only the third one (the genetic algorithm with aggressive mutation) returned the meaningful results that can be used in practice. The classic genetic algorithm in both versions returned results that were not only useless (all the classifiers built over the feature sets returned by the algorithm were overfitted) but also not very practical (because of too large number of the chosen features). Therefore, the final conclusion of the paper is that when a high reduction rate is expected during the feature selection, not only the fitness function but also other algorithm elements have to be redefined.

REFERENCES

[1] Wang, Y., Shangkai, G., Xiaorong, G.: Common Spatial Pattern Method for Channel Selection in Motor Imagery Based Brain-Computer Interface, Proceedings of the 2005 IEEE Engineering in Medicine and Biology 27th Annual Conference (2005)

[2] Oja, E., Yuan, Z.: The FastICA Algorithm Revisited: Convergence Analysis, IEEE Transaction on Neural Networks, (2006), vol. 17, No. 6, pp. 1370-1381

[3] Rejer, I., Górski, P.: Independent Component Analysis for EEG data preprocessing - algorithms comparison, Proceedings of the 12th International Conference on Computer Information Systems and Industrial Management Applications CISIM 2013, Springer (2013)

[4] Kittler, J.: Feature set search algorithms, Pattern recognition and signal processing, (1978), pp. 41--60

[5] Rejer, I.: EEG Feature Selection for BCI Based on Motor Imaginary Task, Foundations of Computing and Decision Sciences, (2012), vol. 37, no. 4, pp. 283--292

[6] Kononenko, I.: Estimating attributes: analysis and extensions of Relief, In: L. De Raedt and F. Bergadano (eds.): Machine Learning: ECML-94, Springer Verlag, (1994), pp. 171--182

[7] Hall, M.A.: Correlation-based feature selection for machine learning, The University of Waikato (1999)

[8] Garrett, D., Peterson, D.A., Anderson, Ch., Thaut, M.H.: Comparison of Linear, Nonlinear, and Feature Selection

Methods for EEG Signal Classification, IEEE Transactions on Neural Systems and Rehabilitation Engineering, (2003), vol. 11, no. 2, pp. 141-145

[9] Tibshirani, R.: Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society. Series B (Methodological), pp. 267--288, vol. 58, no. 1, JSTOR (1996)

[10] Burduk, R.: Recognition Task with Feature Selection and Weighted Majority Voting Based on Interval-Valued Fuzzy Sets in Computational Collective Intelligence. Technologies and Applications, Lecture Notes in Computer Science, ed. Nguyen, Ngoc-Thanh and Hoang, Kiem and Jędrzejowicz, Piotr, Springer Berlin Heidelberg (2012)

[11] Peterson, D.A., Knight, J.N., Kirby, M.J., Anderson, Ch.W., Thaut, M.H.: Feature Selection and Blind Source Separation in an EEG-Based Brain-Computer Interface, EURASIP Journal on Applied Signal Processing, 19, (2005), pp. 3128-3140

[12] Yom-Tov, E., Inbar, G.F.: Feature Selection for the Classification of Movements From Single Movement-Related Potentials, IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 10, no. 3, pp. 170-177 (2002)

[13] Baum, E. B., Boneh, D., Garrett, C.: Where genetic algorithms excel, Evolutionary Computation, (2001), vol. 9, no. 1, pp. 93–124

[14] Kalyanmoy, D., Pratap, A., Sameer, A., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation, IEEE, (2002), vol. 6, no. 2, pp. 182--197

[15] Kołodziej, M., Majkowski, A., Rak, R.J.: A new method of EEG classification for {BCI} with feature extraction based on higher order statistics of wavelet components and selection with genetic algorithms, Adaptive and Natural Computing Algorithms, Springer (2011), pp. 280--289

[16] http://bbci.de/competition/ii/index.html; data set III, II BCI Competition, motor imaginary

[17] Rejer, I.: Genetic Algorithms in EEG Feature Selection for the Classification of Movements of the Left and Right Hand, Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, Advances in Intelligent Systems and Computing, Springer (2013), Vol. 226, pp. 579-589

[18] Lakany, H, Conway, B.: Understanding intention of movement from electroencephalograms, Expert Systems 24 (2007), 5:295--304

[19] Schlögl, A, Lee, F, Bischof, H, Pfurtscheller, G.: Characterization of four-class motor imagery EEG data for the BCI-competition 2005, Journal of neural engineering, IOP Publishing 2 (2005), 4, L14--L22

[20] Raudys, S.J., Jain, A.K.: Small sample size effects in statistical pattern recognition: Recommendations for practitioners. IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (1991), 3, 252–264

Authors

dr hab. nt.. Izabela Rejer, West Pomeranian University of Technology, Szczecin, Żołnierska 49, 71-210 Szczecin, E-mail: irejer@wi.zut.edu.pl; mgr inż. Krzysztof Lorenz, West Pomeranian University of Technology, Szczecin, Żołnierska 49, 71-210 Szczecin, E-mail: klorenz@wi.zut.edu.pl