First approach to solve linear system of equations by using Ant Colony Optimization

Kamil Książek
Faculty of Applied Mathematics
Silesian University of Technology
Gliwice, Poland
Email: kamiksi862@student.polsl.pl

***Abstract*—This paper illustrates first approach to solve linear system of equations by using Ant Colony Optimization (ACO). ACO is multi-agent heuristic algorithm working in continuous domains. The main task is checking efficiency of this method in several examples and discussion about results. There will be also presented future possibilities regarding researches.**

***Index Terms*—linear system of equations, metaheuristics, Ant Colony Optimization, analysis of heuristic algorithm**

I. Introduction

Computers help people to perform complex calculations. They significantly reduce the time required to obtain results and they make not mistakes. There are various aspects of possible applications. Mainly we want computers to process information [1], [2], operate on automated systems, and help people, like in devoted systems for AAL environments [4], [5], [7]. Common usage of computing powers is to process graphics to detect objects [6], assist in voice processing for secure communication [8], help on extraction of important features [20] and improve images [16].

Another possibility to use computing power is solving systems of equations. In practice, engineers often have to deal with this problem. Then very important is proper speed and precision of solutions. Usually, to solve such systems are used numerical methods. This paper attempts to use heuristic algorithms, specifically Ant Colony Optimization to solve linear systems of equations. It has been checked performance of this algorithm using few examples. Then results were discussed.
System of equations were the subject of research many authors. Some information about it is in [3], [10] and [11].

Section II gives information about linear system of equations. In section III is presented description of Ant Colony Optimization with pseudocode. Section IV shows results and discussion about it. Finally, it will be presented possibilities further studies.

II. Linear system of equations

Consider the following system:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = b_n \end{cases} \tag{1}$$

where
$a_{ij} \in \mathbb{R}, \quad b_i \in \mathbb{R}, \quad i,j \in 1,...,n.$

or in the matrix form:

$$A \cdot X = B, \tag{2}$$

where

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix},$$

$$X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \end{pmatrix}^T,$$

$$B = \begin{pmatrix} b_1 & b_2 & \dots & b_n \end{pmatrix}^T.$$

It is assumed that A has nonzero determinant - the system has a one unique solution.

III. Ant Colony Optimization

Ant Colony Optimization (ACO) is a multi-agent heuristic algorithm created for finding global minimum of a function. The inspiration for this method was behaviour of real ant colony. ACO was created by M. Dorigo to solving combinatorial problems [12]. M. Duran Toksari developed Dorigo's algorithm - he invented a method based on ACO to solving continuous problems [13].

The inspiration for this algorithm is the behaviour of ant colony during searching food. Ants have a specific method to communication. They leave chemical substance called pheromone. This allows them to efficiently move - ants can choose shorter path to the aim. The probability of choice the way which has more quantity of pheromone is greater - it means that many ants chose already this road. Following ants reinforce pheromone trace on more efficient track while pheromone is evaporated on the unused path.
Other information about ACO is also presented in [14] while another approach to using ant system in continuous domain is in [15].

IV. Implementation

Algorithm 1:

Ant Colony Optimization Pseudocode of Ant Colony Optimization

Input: number of ants: *m*, number of iteration inside: *s*, number of iteration outside: *W*, boundary of the domain, initial coefficients: α, λ

Output: coordinates of minimum, value of fitness function

Initialisation:
Creating the initial colony of ants.
Searching x_{best} in initial colony; $x_{opt} = x_{best}$.
Calculations:
$i = 1$
while $i < W$ **do**
  $j = 1$
  while $j < s$ **do**
    Moving the nest of ants - defining new territory of ant colony.
    Searching x_j^{best} in present colony.
    if x_j^{best} is better than x_{opt} **then**
      $x_{opt} = x_j^{best}$
    end if
  end while
  Defining new search area (narrowing of the territory).
end while
end

The Algorithm 1 presents the pseudocode of Ant Colony Optimization. The first step is creating m random vectors (ants) filled values from the given domain. Then is necessary to note the quality of these solutions by using fitness function:

$$\Phi(x) = \sum_{i=1}^{n} |b_i - x_i| \tag{3}$$

The function Φ is the sum of errors in all equations of the system. Of course the best values are close to zero. The best from temporary solutions is saved (called x_{best}) and it is provisional place for nest. In this moment x_{best} is also the best solution during the whole search: $x_{opt} = x_{best}$. x_{opt} is the base for next searching step. The successive stage is modifying each coordinate of all vectors according to following formula:

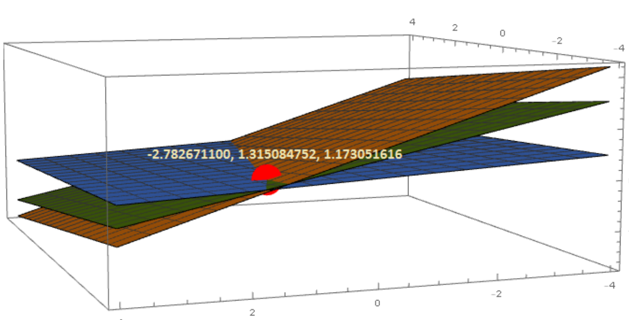

Fig. 1. The graphic interpretation of (7)

$$\forall k \in 1, ..., n \quad x_j^k = x_{opt} + dx, \tag{4}$$

where
j – number of current iteration,
k – number of vector (solution),
$dx = \left(dx^1, dx^2, ..., dx^n\right)$ – vector of pseudorandom values, $dx^i \in [-\alpha_j, \alpha_j]$.

(4) means that $k-th$ vector during $j-th$ iteration is the sum of the best solution (up to $j-1$ iterations, actually it is x_{opt}) and pseudorandom value from the given neighbourhood. The algorithm is checked if $\Phi(x_j^{best}) > \Phi(x_{opt})$, where x_j^{best} is the best solution from j iteration. If the answer is positive, $x_{opt} = x_j^{best}$. This step is carried out s times - there are s internal iterations. If at least one of new points is better approximation of root, it is saved (x_{opt}). The next step is changing the quantity of pheromone - next solutions should be centered around x_{opt}. The main purpose of ACO is narrowing area to search. First steps are in charge of exploration of domain - ants seek promising territory on the whole domain. Next steps are responsible for exploitation (making solutions more precise). α is core value - it is the current quantity of pheromone. The value of α determines area to search. The domain is reducing according to following formula:

$$\alpha_j = \lambda \cdot \alpha_{j-1}, \qquad \lambda \in (0; 1), \tag{5}$$

where
j – number of current iteration.

The value of λ depends on domain. If the domain is relatively wide, λ should be equal more than 0.5 - ants should have more time to find promising territory. In the case narrow domain $\lambda \approx 0.1$ should be sufficient. Searching is continued $W-1$ times with new values of coefficients - there are W external iterations. During following iterations length of the jump is decreased so solutions would be more accurate.

V. Results

A. Tested systems

The benchmark test was carried out by using Ant Colony Optimization on following three linear systems (coefficients were chosen randomly):

1) First system (two equations):

$$A_1 = \begin{pmatrix} 55.09730344 & 38.12917026 \\ 10.57737989 & 86.52430487 \end{pmatrix},$$

$$X_1 = \begin{pmatrix} x_1 & x_2 \end{pmatrix}^T,$$

$$B_1 = \begin{pmatrix} 31.65546153 & 84.06852453 \end{pmatrix}^T,$$

$$A_1 \cdot X_1 = B_1.$$

Fig. 1 shows the graphic interpretation of 1) while Tab. I presents all measurements.

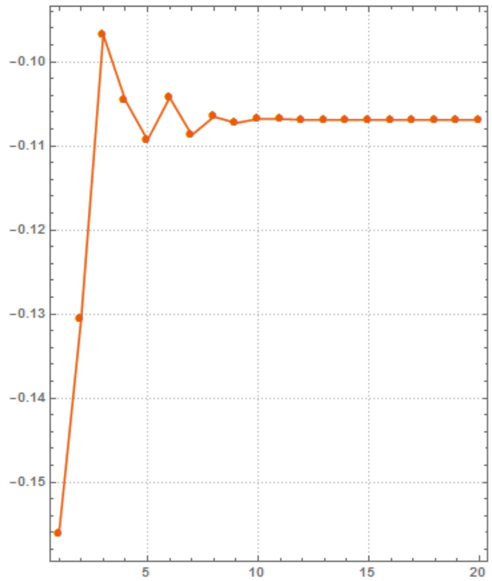

Fig. 2. Result no. 3: value of first coordinate during iterations

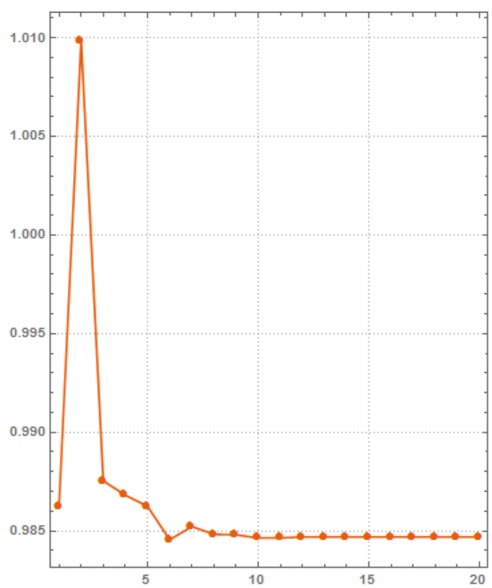

Fig. 3. Result no. 3: value of second coordinate during iterations

2) Second system (three equations):

$$A_2 = \begin{pmatrix} 33.22500927 & 98.10036269 & 26.933598 \\ 49.53447496 & 53.4757128 & 93.1733063 \\ 45.2877606 & 92.67728226 & 38.71016574 \end{pmatrix},$$

$$X_2 = \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix}^T,$$

$$B_2 = \begin{pmatrix} 68.15051868 & 41.78404012 & 41.26656061 \end{pmatrix}^T,$$

$$A_2 \cdot X_2 = B_2.$$

Fig. 2 demonstrates three planes with the point of intersection

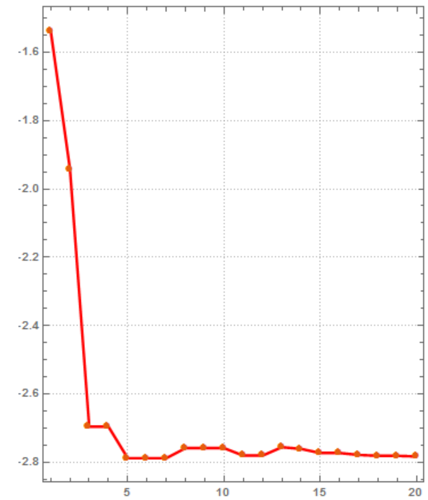

Fig. 4. Result no. 6: value of first coordinate during iterations

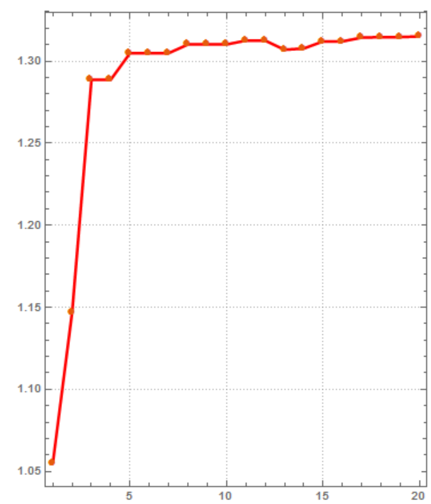

Fig. 5. Result no. 6: value of second coordinate during iterations

– it is 2), while Tab. II shows all results.

3) Third system (four equations):

$$A_3 = \begin{pmatrix} 70.5284618 & 10.71763084 & 84.66285422 & 99.2134024 \\ 83.10581887 & 13.42679705 & 47.42381202 & 90.7001626 \\ 17.26132135 & 71.21872468 & 74.90622771 & 0.7129543228 \\ 1.882180385 & 80.30586823 & 5.591496761 & 98.8264739 \end{pmatrix},$$

$$X_3 = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix}^T,$$

$$B_3 = \begin{pmatrix} 38.17676472 & 87.90315455 & 30.93470124 & 54.32679192 \end{pmatrix}^T,$$

$$A_3 \cdot X_3 = B_3.$$

All measurements from 3) are presented in Tab. III.

B. Discussion

The main advantage of this approach is universality. It is not necessary to transform the system of equations to ensure

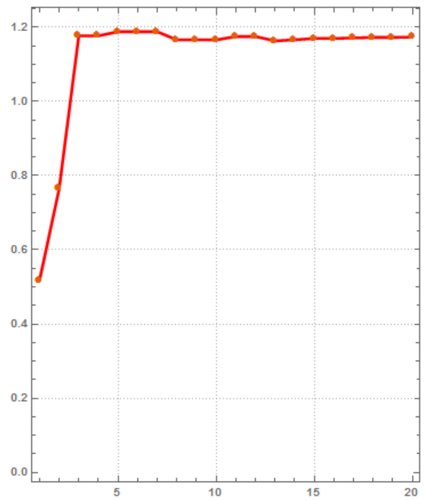

Fig. 6. Result no. 6: value of third coordinate during iterations

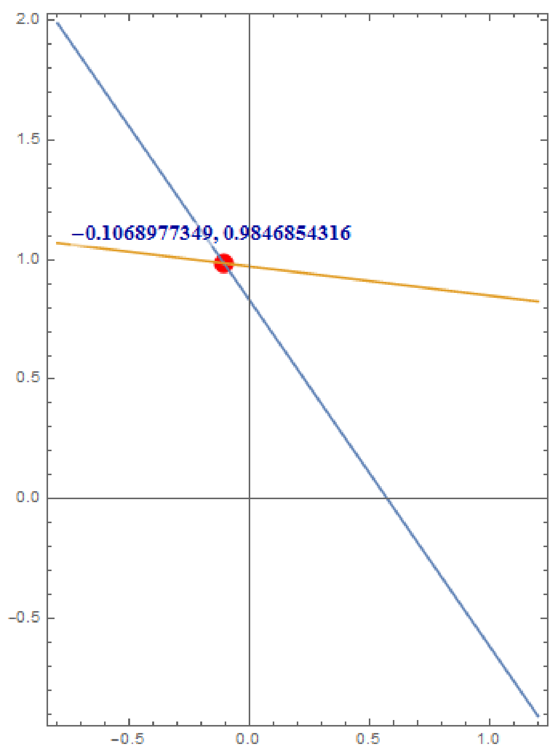

Fig. 7. The graphic interpretation of 1)

convergence (this step is essential in some numerical methods). Results for $x_i, i \in 1, ..., 4$ are presented after rounding. The most important information is value of fitness function. It can be noticed that exactness of results is great. In the case of system of two equations $\lambda = 0.1$ causes that $\Phi(x) = 0$. Fig. 3-4 present values of coordinates during following iterations in the case $\lambda = 0.5$. The graphs illustrate how ACO works. Through initial few iterations values are hesitating and with decreasing α the algorithm is stabilizing around optimal result.

Accuracy of results for the system of three equations was the highest for $\lambda = 0.7$: $\Phi(x) = 0.0160028$. This process is shown on Fig. 5-7. In the case of system of four equations the most effective was $\lambda = 0.4$: $\Phi(x) = 2.3978 \cdot 10^{-6}$.
It is necessary to see that the number of iteration was relatively small. The results would be improved by manipulating initial value of α, value of λ or number of iteration. It is possible to say that approximation in the studied cases is satisfactory.

VI. Conclusions

This paper presents first approach to solve linear systems of equations by using heuristic method (strictly speaking Ant Colony Optimization). There was analyzed three systems (with 2, 3 and 4 variables). This method can be developed in the future. First of all, one can try to use heuristic algorithms to nonlinear systems of equations. There exist less numerical methods to this kind of tasks so heuristic methods may be useful. It is possible to apply some modifications for instance ACO with Local Search or other hybrid algorithm. This topic will be expanded and improved.

References

[1] P. Artiemjew, Stability of Optimal Parameters for Classifier Based on Simple Granules of Knowledge, *Technical Sciences*, vol. 14, no. 1, pp. 57-69. UWM Publisher, Olsztyn 2011.

[2] P. Artiemjew, P. Gorecki, K. Sopyła, Categorization of Similar Objects Using Bag of Visual Words and k – Nearest Neighbour Classifier, *Technical Sciences*, vol. 15, no.2, pp. 293-305, UWM Publisher, Olsztyn 2012.

[3] D. R. Kincaid, and E. W. Cheney. “Numerical analysis: mathematics of scientific computing. Vol. 2.“. American Mathematical Soc. (2002).

[4] R. Damasevicius, M. Vasiljevas, J. Salkevicius, M. Woźniak, “Human Activity Recognition in AAL Environments Using Random Projections” *Comp. Math. Methods in Medicine*, vol. 2016, pp. 4073584:1–4073584:17, 2016, DOI: 10.1155/2016/4073584.

[5] R. Damasevicius, R. Maskeliunas, A. Venckauskas, M. Woźniak, “Smartphone User Identity Verification Using Gait Characteristics” *Symmetry*, vol. 8, no. 10, pp. 100:1–100:20, 2016, DOI: 10.3390/sym8100100.

[6] D. Połap, M. Woźniak, “Flexible Neural Network Architecture for Handwritten Signatures Recognition” *International Journal of Electronics and Telecommunications*, vol. 62, no. 2, pp. 197–202, 2016, DOI: 10.1515/eletel-2016-0027.

[7] D. Połap, M. Woźniak, “Introduction to the Model of the Active Assistance System for Elder and Disabled People,” in *Communications in Computer and Information Science*, vol. 639, pp. 392–403, 2016, DOI: 10.1007/978-3-319-46254-7_31.

[8] D. Połap, “Neuro-heuristic voice recognition,” in *2016 Federated Conference on Computer Science and Information Systems, FedCSIS 2016, Proceedings*. 11-14 September, Gdańsk, Poland, IEEE 2016, pp. 487–490, DOI: 10.15439/2016F128.

[9] C. Napoli, G. Pappalardo, E. Tramontana, Z. Marszałek, D. Połap, M. Wozniak, “Simplified firefly algorithm for 2d image key-points search,” in *IEEE Symposium on Computational Intelligence for Human-like Intelligence*, IEEE 2014, pp. 1–8, DOI: 10.1109/CIHLI.2014.7013395.

[10] J. R. Rice, “Numerical Methods in Software and Analysis“. Elsevier (2014).

[11] A. Ralston, P. Rabinowitz, “A First Course in Numerical Analysis“. Courier Corporation (2001).

[12] M. Dorigo, V. Maniezzo, and A. Colorni. “The ant system: An autocatalytic optimizing process“. (1991).

[13] M. Duran Toksarı, “A heuristic approach to find the global optimum of function.“ Journal of Computational and Applied Mathematics 209.2 (2007): 160-166.

[14] M. Dorigo, T. Stützle, “Ant Colony Optimization“, *MIT Press, Cambridge, MA (2004).*

Table I
RESULTS - SYSTEM OF 2 EQUATIONS

Precise solution: -0.1068977349, 0.9846854316

number	number of points	domain	number of inside iteration	number of outside iteration	initial value of α	λ	x_1	x_2	value of the fitness function
1)	30	$[-5;5]$	20	20	2	0.8	**-0.107503**	**0.984813**	0.033121
2)	30	$[-5;5]$	20	20	2	0.7	**-0.106897**	**0.9847**	0.00184454
3)	30	$[-5;5]$	20	20	2	0.5	**-0.106898**	**0.984685**	$5.79085 \cdot 10^{-6}$
4)	30	$[-5;5]$	20	20	2	0.4	**-0.106898**	**0.984685**	$2.05804 \cdot 10^{-7}$
5)	30	$[-5;5]$	20	20	2	0.1	**-0.106898**	**0.984685**	0.

Table II
RESULTS - SYSTEM OF 3 EQUATIONS

Precise solution: -2.782671100, 1.315084752, 1.173051616

number	number of points	domain	number of inside iteration	number of outside iteration	initial value of α	λ	x_1	x_2	x_3	value of the fitness function
6)	30	$[-5;5]$	20	20	2	0.7	**-2.7831**	**1.31511**	**1.17331**	0.0160028
7)	30	$[-5;5]$	20	20	2	0.5	**-2.82615**	**1.32503**	**1.19046**	0.37346
8)	30	$[-5;5]$	20	20	3	0.5	**-2.77983**	**1.31443**	**1.17191**	0.0244493
9)	30	$[-5;5]$	20	20	2	0.45	**-2.78489**	**1.31559**	**1.17394**	0.0190372
10)	30	$[-5;5]$	20	20	2	0.4	**-2.82968**	**1.32584**	**1.19187**	0.403732
11)	30	$[-5;5]$	20	20	2	0.3	**-2.77648**	**1.31359**	**1.17062**	0.0541228
12)	30	$[-5;5]$	20	20	2	0.2	**-2.7574**	**1.30892**	**1.16316**	0.221745
13)	30	$[-5;5]$	20	20	2	0.1	**-2.90849**	**1.34566**	**1.22239**	1.10246

Table III
RESULTS - SYSTEM OF 4 EQUATIONS

Precise solution: 1.486007266, 0.8550637392, -0.7411742368, -0.1314678539

number	number of points	domain	number of inside iteration	number of outside iteration	initial value of α	λ	x_1	x_2	x_3	x_4	value of the fitness function
14)	30	$[-5;5]$	20	20	2	0.8	**1.4945**	**0.858514**	**-0.746341**	**-0.133945**	0.352619
15)	30	$[-5;5]$	20	20	2	0.7	**1.48602**	**0.854945**	**-0.741245**	**-0.131361**	0.0240107
16)	30	$[-5;5]$	20	20	2	0.6	**1.486**	**0.855079**	**-0.741185**	**-0.131464**	0.00293356
17)	30	$[-5;5]$	20	20	2	0.5	**1.48601**	**0.855063**	**-0.741174**	**-0.131468**	0.0000864998
18)	30	$[-5;5]$	20	20	2	0.45	**1.48601**	**0.855064**	**-0.741174**	**-0.131468**	0.0000174678
19)	30	$[-5;5]$	20	20	2	0.4	**1.48601**	**0.855064**	**-0.741174**	**-0.131468**	$2.3978 \cdot 10^{-6}$
20)	30	$[-5;5]$	20	20	2	0.3	**1.47847**	**0.852423**	**-0.736947**	**-0.129418**	0.276773
21)	30	$[-5;5]$	20	20	2	0.2	**1.51865**	**0.874344**	**-0.766887**	**-0.146302**	1.54667

[15] K. Socha, M. Dorigo, "Ant colony optimization for continuous domains.“ European journal of operational research 185.3 (2008): 1155-1173.

[16] M. Woźniak, "Novel Image Correction Method Based on Swarm Intelligence Approach," in *Communications in Computer and Information Science*, vol. 639, pp. 404–413, 2016, DOI: 10.1007/978-3-319-46254-7_32.

[17] M. Woźniak, D. Połap, M. Gabryel, R.K. Nowicki, C. Napoli, E. Tramontana, " Can we process 2d images using artificial bee colony?". In *International Conference on Artificial Intelligence and Soft Computing*, pp. 660-671, 2015.

[18] D. Połap, M. Woźniak, , C. Napoli, and E. Tramontana, "Is swarm intelligence able to create mazes?" *International Journal of Electronics and Telecommunications*, vol. 61, No. 4, pp. 305–310, 2015, DOI: 10.1515/eletel-2015-0039.

[19] D. Połap, M. Woźniak, , C. Napoli, and E. Tramontana, "Real-Time Cloud-based Game Management System via Cuckoo Search Algorithm" *International Journal of Electronics and Telecommunications*, vol. 61, No. 4, pp. 333–338, 2015, DOI: 10.1515/eletel-2015-0043.

[20] M. Woźniak, D. Połap, C. Napoli, and E. Tramontana, "Graphic object feature extraction system based on Cuckoo Search Algorithm" *Expert Systems with Applications*, vol. 60, pp. 20–31, 2016, DOI: 10.1016/j.eswa.2016.08.068.

[21] L. Zmudzinski, A. Augustyniak, P. Artiemjew, Control of Mindstorms NXT robot using Xtion Pro camera skeletal tracking, *Technical Sciences*, vol. 19, no.1, pp. 71-81, Olsztyn, UWM Publisher, 2016.

[22] K. Trojanowski, "Metaheurystyki praktycznie“, Warsaw School of Information Technology, p.7, Warsaw 2008.