PERFORMANCE COMPARISON OF OPTIMIZATION ALGORITHMS FOR THE DESIGN OF ROBOTS

ANISIA NEAMTIU[1], VETURIA CHIROIU[2], CATALIN BOANTA[3], CORNEL BRISAN[4]

Abstract. The design of robots is a complex task, that requires to identify the most suitable architecture of a robot that satisfies specific criterions. In such cases, optimization methods are used in order to identify the optimal values of certain parameters that describe the architecture of a robot that satisfies the criterions or lead to high kinematic, static or dynamic performance. The optimization methods are implemented using several optimization algorithms, that have to be chosen according to the type of the involved parameters, the cost functions and the imposed constraints. The results of an optimization are influenced by the type of the implemented algorithm, regardless of its type. This is why, for the same optimization problems, two or more algorithms may achieve different results and require different computational time and resources. Consequently, the problem addressed in this paper is represented by the comparison of several optimization algorithms, in order to identify their degree of efficiency and efficacy for optimization of robots, taking into considerations different performance criterions.

Key words: optimization, algorithm, methods, performance, comparison, robot, efficiency, efficacy.

1. INTRODUCTION

Optimization represents the action of identifying the best result or the result that fulfills certain criterions of a specific problem. This action may be realized based on a previous experience of a human designer or may be implemented as a mathematical function, being necessary to identify the optimal solutions.

The first mathematical optimization methods have been proposed by Newton, who has developed iterative methods to identify the optimal solutions. Later on, Lagrange and Fermat have developed calculus-based methods.

[1] Technical University of Cluj-Napoca, Romania, email Anisia.Neamtiu@gmail.com

[2] Institute of Solid Mechanics, Romanian Academy, Bucharest, Romania, email: veturiachiroiu@yahoo.com

[3] Technical University of Cluj-Napoca, Romania, email: Catalin.Boanta02@gmail.com

[4] Technical University of Cluj-Napoca, Romania, email: Cornel.Brisan@mdm.utcluj.ro, corresponding author

During the industrial revolutions, the demand of optimization methods has emerged, due to the need of development of new technical equipment and the necessity of increasing the performance of the already developed ones.

Still, the growth of the computational power from the last three decades has led to a real interest in implementation of optimization algorithms. Therefore, nowadays, optimization algorithms are used in order to identify optimal solutions of real problems, being applied in mechanical or electrical engineering, economics, robotics and genetics (the applications are not limited to these domains).

With regard to the design of robots, optimization is used to identify the optimal values of certain parameters that describe the architecture of a robot that satisfies certain criterions or lead to high performance regarding the kinematics, statics or dynamics. Optimization methods are implemented using different algorithms, that have to correspond to the type of the parameters, the cost functions and the imposed constraints. Due to the high complexity of the functions that describe the architecture of a robot, most of the used algorithms are stochastic, that involve probabilistic data in order to achieve the optimal solutions. The results of an optimization are influenced by the type of the implemented algorithm, regardless of its type. This is why, for the same optimization problems, two or more algorithms may achieve different results and require different computational time and resources. Based on these aspects, the scientific problem addressed in this paper is represented by the comparison of several optimization algorithms, in order to identify their degree of efficiency and efficacy for optimization of robots, taking into considerations different performance criterions. The algorithms are compared both regarding their ability of reaching an optimal value and the required computational time. The paper is structured as follows: the Chapter 2 and the Chapter 3 illustrate the literature review of using the optimization and the optimization algorithms in robotics, the Chapter 4 presents the problem formulation, the Chapter 5 and the Chapter 6 illustrate the algorithm comparison setup and the corresponding numerical results and the Chapter 7 summarizes the paper and presents the conclusions.

2. OPTIMIZATION IN ROBOTICS – GENERAL ASPECTS

Optimization may be defined as the process of identifying of the best result of a specific problem, taking into consideration several constraints, conditions of circumstances that have to be respected. From an engineering point of view, this aspect implies the transformation of a real problem in a mathematical function of one or more variables, that has to be minimized or maximized with respect to the constraints imposed by a human designer or by the problem that is optimized.

An optimization problem may be represented as follows. Find a vector x

$$x = \left(x_1, x_2, \ldots, x_n\right), \tag{1}$$

that satisfies the minimum or maximum condition of a function

$$f = f(x), \tag{2}$$

taking into consideration a number of m inequalities and p equalities constraints

$$\begin{aligned} g_i(x) &\leq 0 \\ k_j(x) &= 0 \end{aligned}, \tag{3}$$

where $i = 1\ldots m$ and $j = 1\ldots p$.

In the above equations, x is called a design vector, f is the cost function (also called objective or fitness function), $g_i(x)$ represent the inequality constraints and $k_j(x)$ represent the equality constraints. An optimization may or not include constraints (also called restrictions) and may include more than one objective functions (in which case, the optimization is called multi-objective).

Depending on the application, the optimization methods may be classified in continuous or discrete, constrained or unconstrained, global or local, linear or nonlinear, single or multi-objective and stochastic or deterministic. More on the classification of the optimization methods may be found in [1–5].

In any robotic industrial application, the robots have to fulfill several specific criterions based on the requirements imposed by the application they are used for. This means that a robot has to respect some performance factors regarding the kinematic, static or dynamic behavior. In order to fulfill those factors, the architecture of the robot has to be designed using optimization methods.

According to [6], in robotics, regardless of the complexity of the performance factors, an optimization method is based on the following steps:

– The identification of the design variables, of the constraints and performance indices imposed by the application;
– The mathematical formulation of the performance indices;
– The formulation of the optimization problem;
– The implementation using an optimization algorithm, searching for the optimal solution and interpretation of the results.

The design variables are the parameters that describe the architecture of a robot (number of elements, geometrical lengths, material, number of actuators), that influence the behavior of the robot given by specific performance indices. These indices are used in order to quantify the performance of the robot with regard to kinematics, or dynamics. In the following, some performance factors are presented.

The workspace of a robot is one of the most common performance factors used for many applications, being defined by the locations/ positions (or the total volume) that may be reached by the end-effector. For a specific application, the workspace has to be maximized or to be as close as possible to an imposed one [7–12]. In [13] and [14] several types of workspaces are presented as: the constant orientation workspace, the orientation workspace, the reachable workspace, or the inclusive orientation workspace.

With regard to the kinematics, the most common terms that measure the kinematics performance presented in the scientific literature are the dexterity, manipulability and isotropy are the most common ones.

The dexterity describes the mobility of a robot within its workspace, defined by the number of directions the points within the workspace may be reached by the end effector of a robot. The orientation of the end-effector is described by the rotation matrix $R(\alpha,\beta,\gamma)$ using roll, pitch and yaw angles α,β,γ within the range $(0,2\pi)$ [15]. The dexterity index $d\in[0,1]$ is defined as

$$d=\frac{d_x+d_y+d_z}{3}, \tag{4}$$

where d_x,d_y,d_z are indices in directions X,Y,Z defined as

$$d_x=-\frac{\Delta\alpha}{2\pi},\ d_y=-\frac{\Delta\beta}{2\pi},\ d_z=-\frac{\Delta\gamma}{2\pi}, \tag{5}$$

where Δ measures a possible variation of the angle for each point in the workspace. The dexterity index is not constant within the workspace.

The manipulability index is evaluated as a function of the configuration of the robot and the values in the active joints, the index reflecting the ability of the robot to arbitrarily change the position and orientation of the end effector. To evaluate the manipulability index, the Jacobian matrix of the robot J has to be calculated, in order to evaluate the mathematical relationship between the generalized coordinates of the active joints q and the pose of the end-effector p as:

$$J\dot{q}=p\,. \tag{6}$$

The manipulability index has been firstly introduced in [16], being evaluated with:

$$\mu=\sqrt{\det(JJ^T)}\,. \tag{7}$$

Also, the manipulability index may be calculated as a product of the singular value of the JJ^T matrix. In the case when a manipulator has a square Jacobian, the manipulability index is

$$\mu=\left|\det(J)\right|. \tag{8}$$

The isotropy index of a manipulator, is evaluated with the condition number of the Jacobian matrix, being an index that illustrates the kinematic performance [17]. For a square Jacobian, the condition number is a measure of Jacobian invertibility and evaluated with

$$k(J) = \| J \| \| J^{-1} \| \ , \ \| J \| = \sqrt{\mathrm{tr}(JWJ)^T} \ , \ W = \frac{1}{\dim(J)} I \ , \tag{9}$$

where I is a unit matrix with the same dimensions as the Jacobian. In the case of a non-square Jacobian matrix, the condition number is

$$k(J) = \frac{\sigma_{\max}}{\sigma_{\min}} , \tag{10}$$

where $\sigma_{\max}$ and $\sigma_{\min}$ are the maximum and minimum singular values of J. The value from the above equation is valid only if the elements of the Jacobian matrix have the same units of measurements contrary a scaling of the matrix is required. The value of the conditioning number is within the range:

$$k(J) \in [0, +\infty) , \tag{11}$$

this aspect being a computational disadvantage (not having an upper bound) This is why, the Local Conditioning Index (LCI) has been introduced as

$$\mathrm{LCI} = \frac{1}{k(J)} , \tag{12}$$

that lies within the range $[0,1]$. The LCI is expressed for a certain pose of the end effector. In order to quantify this value for the entire workspace, as presented in [18], the Global Conditioning Index (GCI) is evaluated as:

$$\mathrm{GCI} = \frac{\int_{WS} \frac{1}{k(J)} \mathrm{d}WS}{\int_{WS} \mathrm{d}WS} . \tag{13}$$

3. OPTIMIZATION ALGORITHMS IN ROBOTICS – STATE OF THE ART

In robotics, an optimization method requires the development of an objective function that quantifies from a mathematical point of view one or more performance factors. Most of the factors (or indices) evaluates the dimensions of the workspace, the kinematic or dynamic performance, trajectory planning or others. These factors present a maximum or minimum value, are evaluated using non/linear equation and, in some cases, do not have a closed form solution. This is why, the optimization algorithm used to find the optima has to be chosen accordingly to the nature of the performance indices. Most of the used algorithms used in the scientific literature are the stochastic ones (that use probabilities or randomly chosen data). In the following, the most common used algorithms in optimization of robots are presented.

Brute Force Optimization refers to the identification of the optimal solution for the objective function by combining all the possible values that may be imposed to the design vector. This algorithm has been used in [19] in order to optimize a surgical robot in order to achieve the desired pose of the end effector. Also, in [20] Brute Force Optimization has been used to maximize the workspace of a robot. Even though this algorithm may lead to optimal solution, as presented in [21], the required computational time to reach an optimum has an exponential growth with the number of design variables. Also, the computational time is increased by the precision of the discretization of the search space [19], [20]. Even though, theoretically, the algorithm may lead to a global optimum, due to the computational restriction, practically it may not be achieved.

The Genetic Algorithm is inspired by the genetics of the living organisms. The algorithm keeps the information of the solutions using codified populations that evolve in time by applying the operators used in genetics: reproduction, crossover and mutation. A new population is created at each iteration, the objective function being evaluated for each member of the population (each combination of the design vector). This process is repeated until a stop criterion is satisfied (a maximum number of generations has been achieved, the objective function has reached a certain value). As presented in [22], the Genetic Algorithms are suitable for non-linear problems, having the advantage of reaching an optimal solution or a nearly optimal solution, with low computational times. This type of algorithms has been used in several recent papers for designing robots. In [23], the algorithm has been used for reaching a certain workspace and maximization of the GCI. Also, in [24] a Genetic Algorithms has been used to optimize a robot, imposing as performance criterions the rigidity, dexterity, manipulability and the maximization of the workspace. In [25], the authors propose a geometric optimization of a DELTA Robot in order to maximize the GCI within the workspace. These papers illustrate the advantages of the Genetic Algorithms as the robustness (the ability to reach an optimum regardless of the nature of the objective function) and the ability to explore a large search space.

The Particle Swarm Optimization is inspired also from nature. This stochastic algorithm simulates the behavior of the bird flocking or fish schooling when moving in order to find food (the location of the food being the optimal solution). This is an evolutionary algorithm, the position of each member of the swarm being updated at each iteration according the previous position and the best position of each individual. This algorithm has been extensively used for optimizing robots. In [26], the authors propose the use of the Particle Swarm Optimization for minimizing the rigidity of a parallel robot within the workspace. Moreover, the author compares the Particle Swarm Optimization with the Genetic Algorithm, illustrating that the first one has better performance. More examples are the papers [27] and [28] in which the Particle Swarm Optimization is used to optimize parallel mechanism taking as performance factors the global compliance or the GCI . In these papers, the authors do not make a reasoning regarding the choice of this specific algorithm. The papers [29] and [30] presents advantages and disadvantages of this algorithm. On the positive side, it is stated that the results are not affected by the initial size of e swarm, the algorithm presenting the ability to converge rapidly to a solution. On the negative side, it is mentioned that highly complex solution are not suited for this algorithm (even though, in [29] is mentioned that it may be used for scientific research or engineering purposes).

The direct search algorithm called Pattern Search is an evolutionary algorithm, with a simple concept, easy to be implemented and efficient regarding its computational time. As presented in [31], it has the ability to identify a global optimum and the possibility to fine search in the vicinity of a local optimum. The algorithm starts at a point in which the objective function is evaluated. Furthermore, it creates a grid of points near the current one in which the cost function is yet again evaluated. The best point in the grid becomes the current point and the process is restarted until an optimum is reached. This algorithm has been implemented for optimization of a serial robots in [32], taking as performance criterions the dexterity and the workspace. Also, the paper [33] presents the optimization of a parallel robots in order to maximize several kinematic indices using both the Pattern Search and the Genetic Algorithm. Both of these algorithms have presented similar efficacy, but the Genetic Algorithm has required a longer computational time.

The Simulated Annealing is an iterative algorithm that uses the annealing principles from metallurgy a material is heated then it is cooled in a controlled manner, in order to decrease its internal energy. In the algorithm, the internal energy of the material is the objective function of the optimization and the temperature is a parameter that is progressively decreased (by using an user imposed function). Even though the Simulated Annealing is a local optimization search method, it has the advantage that it does not converge to a local minimum by accepting, based on a probability function, some solutions less advantageous. By starting from an initial state and an initial optimal solution, the algorithm is searching in its vicinity a new state that has a lower value for the objective function (of the internal energy). If such a state is identified, it becomes the new solution and the new current state. If in the vicinity of the current state there are no new state that lower the objective function, the transition to a one of this is realized based on a probabilistic function that considers the difference between the objective functions of the two states and a temperature parameter. The temperature parameter is initialized with a high value that is lowered on each iteration of the algorithm (similarly to the decrease of the temperature in the real annealing from metallurgy) [34]. The Simulated Annealing has been used in the scientific literature in order to optimize the architecture of the robots, such an example being [35]. In this paper, a serial manipulator is optimized considering the volume of the workspace and the total dimension of the mobile elements. Similarly, in [36] the optimization of a DELTA robot is proposed, having as performance factor the size of the workspace.

With regard to the performance comparison of the presented algorithms, there were few papers that presents a systematic comparison in terms of efficacy (the ability to reach an optimum) and efficiency (the ability to reach an optimum with low computational time) between many types of optimization algorithms used in robotics. Most of the used algorithms are the Particle Swarm Optimization and the Genetic Algorithms. For example, in [37] it is stated that the Genetic Algorithms has performed lower compared to Particle Swarm Optimization when solving the kinematics of a serial robot. This comparison is presented also in [38], in which, in terms of efficacy there was no difference regarding the performance. On the efficacy side, the Particle Swarm Optimization has required lower computational times.

In [39], a robot is optimized with regard to its geometrical dimensions and the size of the workspace, four algorithms being compared, among them being the Genetic Algorithms and the Simulated Annealing. The paper shows that the Simulated Annealing has performed considerably better than the Genetic Algorithm. The same results are presented also in [40]: an increased number of design parameters leads to higher computational times for the Genetic Algorithm.

In [41] a comparison between the Particle Swarm Optimization, the Simulated Annealing, the Genetic Algorithm and the Pattern Search is conducted for optimizing the displacement of pipe robots. Even though, for a similar computational time of one hour, the value of the objective function has been almost similar, the most performant algorithm has been the Particle Swarm Optimization and the least performant one has been the Pattern Search. If the computational time has been reduced to 5 minutes, the Simulated Annealing has obtained the best results and the worst results correspond to the Particle Swarm Optimization.

Another paper that presents a comparison between the Genetic Algorithm and the Pattern Search is [42], the performance factor being a kinematic index. The paper states that the efficacy of the algorithms has been similar but the Pattern Search has been less efficient, requiring higher computational times.

4. PROBLEM FORMULATION

As presented in the previous section in order to implement optimization methods in robotics, some of the most used algorithms are the Particle Swarm

Optimization, the Genetic Algorithms, the Pattern Search and the Simulated Annealing. According to the current state of the art, there are slightly different opinions regarding the better choice among these algorithms, all of these being suitable for optimization methods in robotics (in which non/linear equations are used, a global optimum is preferred, the model may not be fully known in closed-form, and without having the derivatives of the objective functions). Therefore, the scientific problem addressed in this paper is the identification of one or more algorithms that presents the best performances when applied in robotics, for several objective functions, both in terms of efficiency as in efficacy. The first step is to establish the imposed objective functions, the second step is to implement the optimization with all the functions using the chosen algorithms, and the third step is to identify the most suitable algorithms. The problem formulation is presented in the Fig. 1.

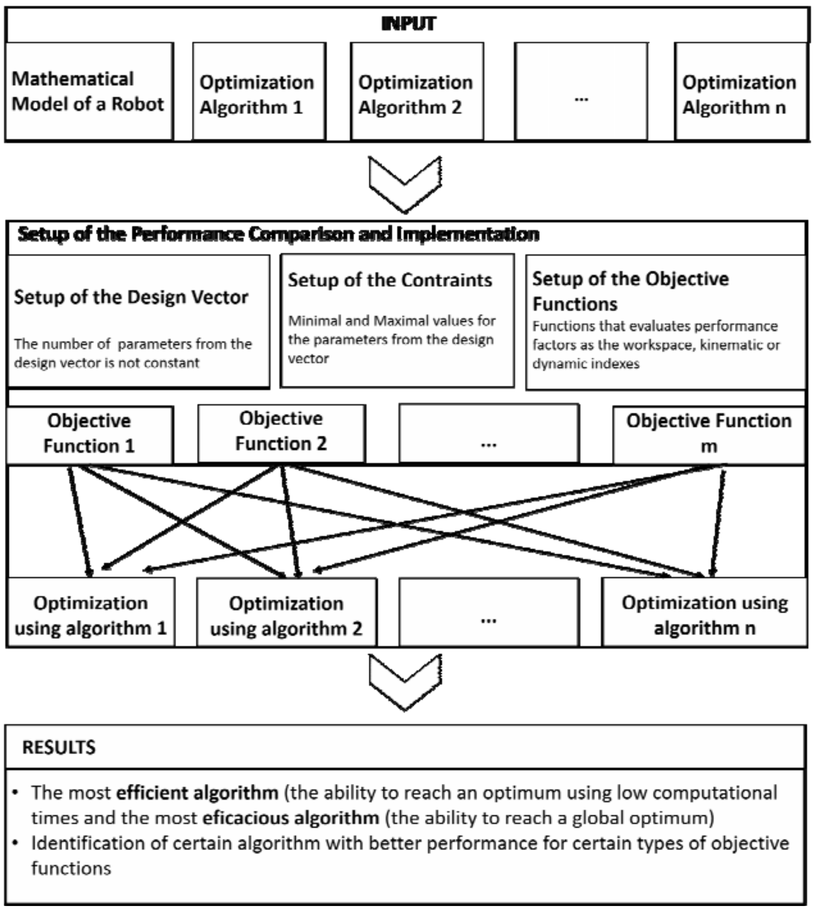
INPUT
Mathematical Model of a Robot
Optimization Algorithm 1
Optimization Algorithm 2
...
Optimization Algorithm n
Setup of the Performance Comparison and Implementation
Setup of the Design Vector
The number of parameters from the design vector is not constant
Setup of the Contraints
Minimal and Maximal values for the parameters from the design vector
Setup of the Objective Functions
Functions that evaluates performance factors as the workspace, kinematic or dynamic indexes
Objective Function 1
Objective Function 2
...
Objective Function m
Optimization using algorithm 1
Optimization using algorithm 2
...
Optimization using algorithm n
RESULTS
• The most efficient algorithm (the ability to reach an optimum using low computational times and the most eficacious algorithm (the ability to reach a global optimum)
• Identification of certain algorithm with better performance for certain types of objective functions

Fig. 1 – Problem formulation.

5. ALGORITHM COMPARISON SETUP

The algorithms that are compared regarding their optimization performance are the Genetic Algorithm, the Particle Swarm Optimization, the Pattern Search and the Simulated Annealing.

In order to compare the performance of the algorithms the following conditions are imposed:

– The number of design variables is 2, 4 or 6;
– The algorithms have run 10 times and the medium and maximal results of the objective function has been considered. Each of this run has considered a maximal number of 10.000 evaluations of the cost function;
– The cost functions are the maximization of the workspace and the maximization of the global conditioning index.

As a prerequisite, in order to compare the algorithms in robotics, the mathematical models of the optimized robots have to be developed. In the case of this paper, the optimized robot is a 6 degrees of freedom parallel robot with rotational actuators, as presented in the Fig. 2 and the Fig. 3.

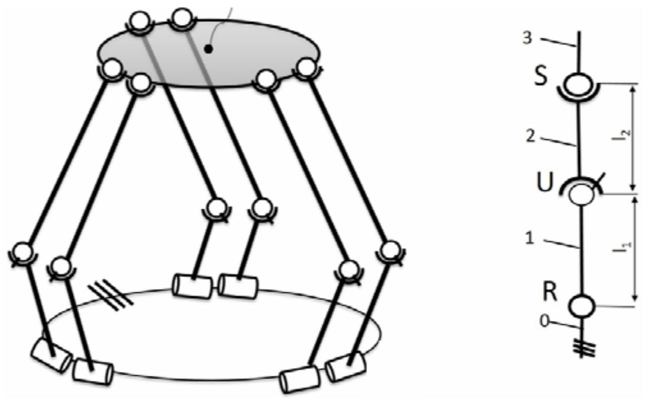

Fig. 2 – 6 DOF Parallel robot with rotational actuators.

The robot is composed by a fixed platform, a mobile platform and 6 rotational-universal-spherical (RUS) kinematic open loops. It is beyond the scope of this paper to present all the mathematical formulations regarding the kinematics of this robot. A thoroughly presentation of the kinematics is presented in [43], and other analysis of kinematics of parallel robots are presented in [44] and [45]. The notations from the Fig. 2 and the Fig. 3 are: l_1 and l_2 are the lengths of the first and second mobile elements of the RUS open loops, R and r are the radiuses of the fixed and mobile platforms α_i and β_i ($i = 1...6$) are the angles of positioning the of the spherical and rotational joints on the fixed and mobile platforms, respectively.

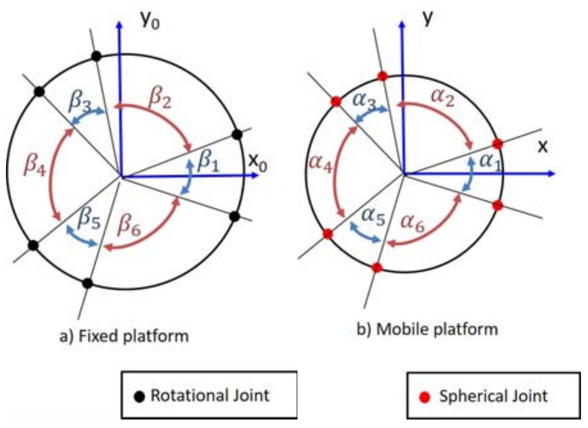

Fig. 3 – Fixed and mobile platforms of the robot.

The implementation has been realized separately for three design vectors with different lengths, with 2, 4 and 6 variables, as presented in the equations below.

$$x_{2vars} = [l_1, l_2], \tag{14}$$

$$x_{4vars} = [l_1, l_2, \text{ratio}_d, \text{ratio}_u], \tag{15}$$

$$x_{6vars} = [l_1, l_2, \text{ratio}_d, \text{ratio}_u, r, R], \tag{16}$$

The constraints of the optimization have been imposed for each length of the design vector. Therefore, for the case of two variables, the constraints presented in a matrix form are:

$$A_{2vars} x_{2vars}{}^T < b_{2vars}, \tag{17}$$

where,

$$A_{2vars} = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 1 & -1 & 0 & 1 \end{bmatrix}^T, \tag{18}$$

and

$$b_{2vars} = [-0.001 \ -0.001 \ 1 \ 1]^T . \tag{19}$$

For the vector with 4 variables, the imposed constraints presented as a matrix form are:

$$A_{4vars} x_{4vars}{}^{T} < b_{4vars}\,, \tag{20}$$

where,

$$A_{4vars} = \begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}^{T}, \tag{21}$$

and

$$b_{4vars} = \begin{bmatrix} -0.001 & -0.001 & 1 & 1 & -0.1 & -0.1 & 10 & 10 \end{bmatrix}^{T}. \tag{22}$$

For the vector with 6 variables, the imposed constraints presented in a matrix form are:

$$A_{6vars} x_{6vars}{}^{T} < b_{6vars}\,, \tag{23}$$

where

$$A_{6vars} = \begin{bmatrix} -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^{T}, \tag{24}$$

and

$$b_{6vars} = \begin{bmatrix} -0.001 & -0.001 & 1 & 1 & -0.1 & -0.1 & 10 & 10 & -0.001 & -0.001 & 0 \end{bmatrix}^{T}. \tag{25}$$

The four algorithms have been tested for optimizing separately two objective functions, commonly used in optimization problems of robots. The first optimization criterion is the maximization of the dimension of the workspace. Considering that the workspace is discretized in points, the dimensions of the workspace is regarded as the total number of points M included in the workspace:

$$Fit\_WS = M\,. \tag{26}$$

The second objective function the algorithms have been tested with is the maximization of the Global Conditioning Index, GCI, presented in the eq. (13):

$$Fit\_GCI = \frac{\int_{WS} \frac{1}{k(J)} dWS}{\int_{WS} dWS} \cong \frac{1}{M} \sum_{j=1}^{M} \frac{1}{\| J_j \| \| J_j^{-1} \|} . \tag{27}$$

6. NUMERICAL RESULTS

The numerical results from this section illustrates the performance of each algorithm in order to optimize the architecture of a robot, considering the design vector with 2, 4 and 6 variables and two different objective functions. Each algorithm has run a number of 10 000 iterations for each cost function.

The results obtained by the Genetic Algorithm when optimizing a robot considering as cost function the workspace and the GCI are presented in the Fig. 4 and the Fig. 5

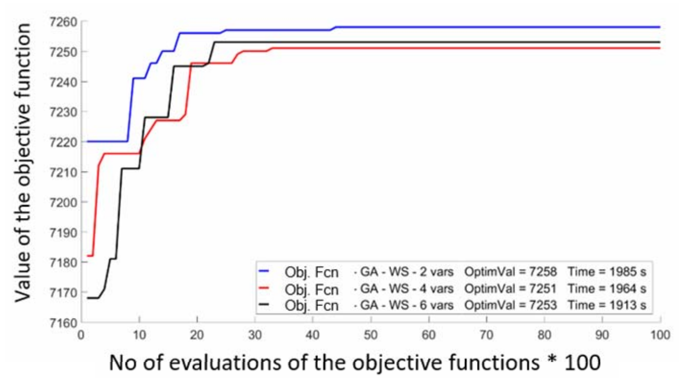

Fig. 4 – Convergence of the Genetic Algorithm for the workspace.

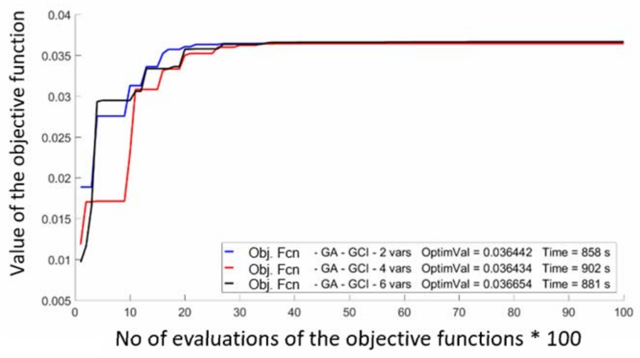

Fig. 5 – Convergence of the Genetic Algorithm for the Global Conditioning Index.

Analyzing the previous two figures, it can be observed that the value of the objective function for the Genetic Algorithm is strongly modified in the first 3000 iterations. After this, the value is stabilized, the optimal value being reached around the iteration 4 000. Also, according to the graphics, an increase of the number of design variables does not lead to a higher optimal value. In both figures, the higher value is corresponding to the 2 variable case, followed by the 6 variable and 4 variables cases.

In the case when the used algorithm is the Particle Swarm Optimization, the results of the optimization for the same objective functions are presented in the Fig. 6 and Fig. 7.

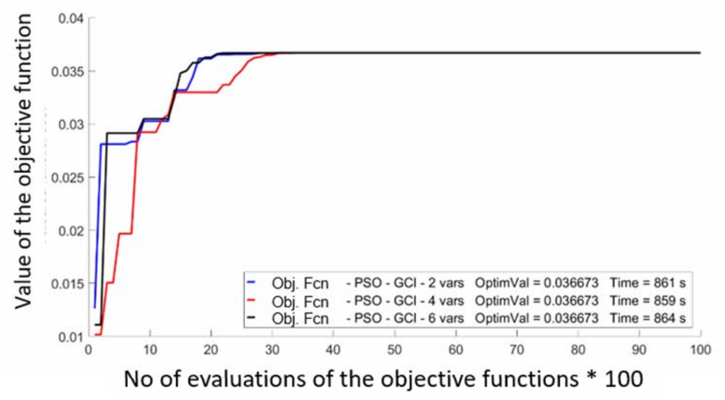

Fig. 6 – Convergence of the Particle Swarm Optimization for the workspace.

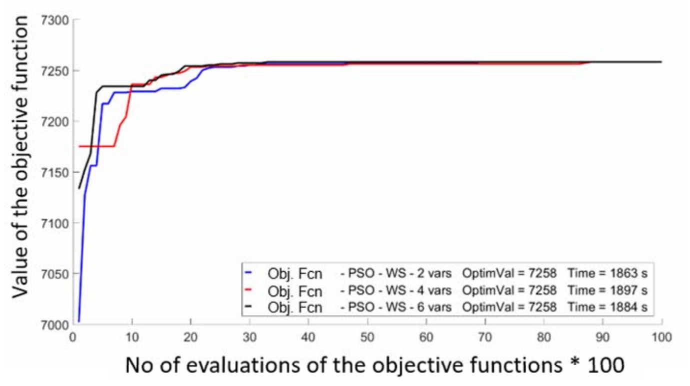

Fig. 7 – Convergence of the Particle Swarm Optimization for the Global Conditioning Index.

By analyzing the previous two figures, in a similar way as for the Genetic Algorithm, the cost function in the case of the Particle Swarm Optimization is strongly varied in the first 3000 iteration. On the other hand, the maximal value of the cost function is reached regardless of the number of variables in the design vector (two, four or six variables).

The results of the optimization for the Pattern Search algorithm are presented in the Fig. 8 and Fig. 9.

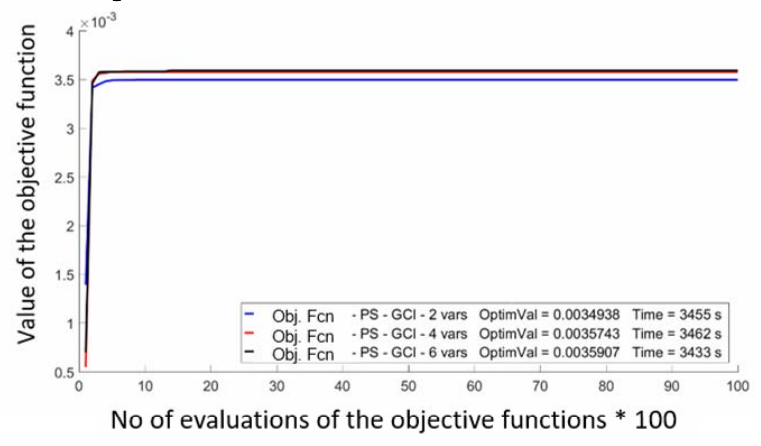

Fig. 8 – Convergence of the Pattern Search for the workspace.

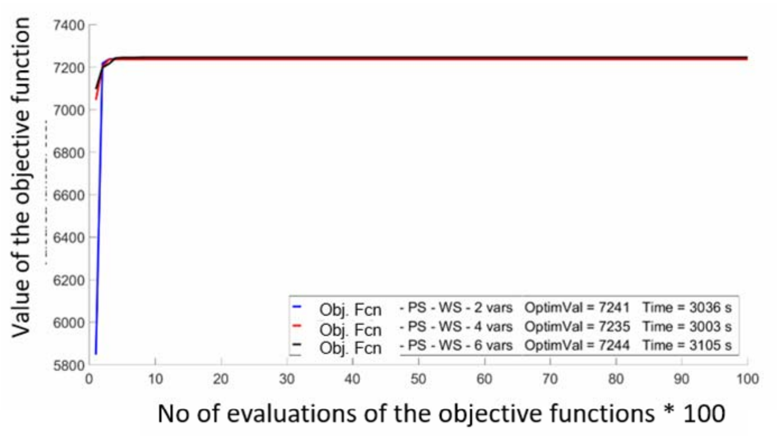

Fig. 9 – Convergence of the Pattern Search for the Global Conditioning Index.

The Fig. 8 and the Fig. 9 show that the Pattern Search presents a very fast convergence, reaching the optimal value in most of the cases after 300 iteration (ten times faster than the Genetic Algorithm or the Particle Swarm Optimization). Still, the fast convergence leads to a mean value of the cost function considerably lower than in the case of the first two algorithms, that may be reflected to the fact that Pattern Search reaches a local optimum. In the case when the cost function is the GCI, the values obtained by the Pattern Search are lower with an order of magnitude as for the first two algorithms. On what regards the used number of variables, the Pattern Search has reached higher values by using a larger number of variables in the design vector.

The results of the optimization in the case when the used algorithm is the Simulated Annealing are presented in the Fig. 10 and Fig. 11.

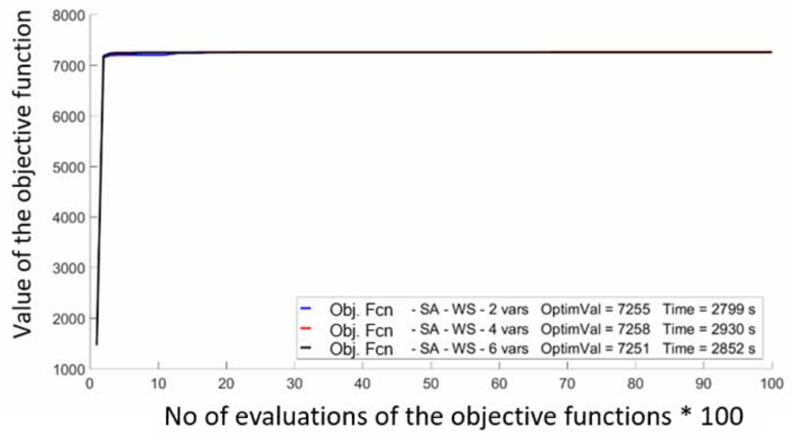

Fig. 10 – Convergence of the Simulated Annealing for the workspace.

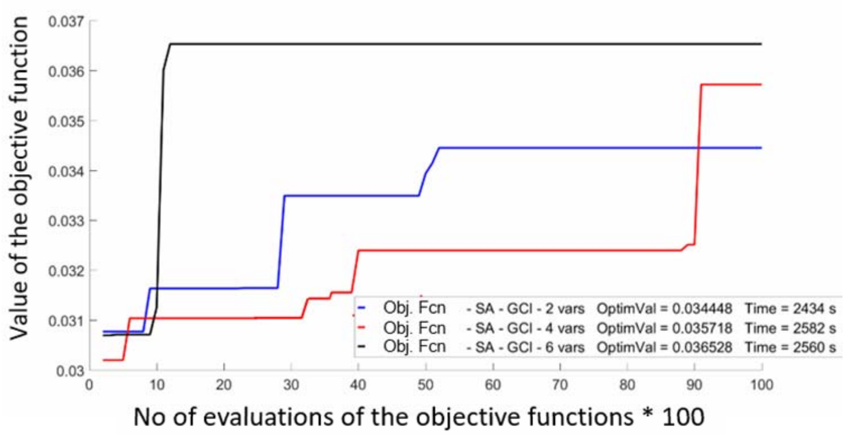

Fig. 11 – Convergence of the Simulated Annealing for the Global Conditioning Index.

The convergence of the Simulated Annealing is different for the two objective functions. For the function that evaluates the workspace (Fig. 10), the algorithm presents a fast convergence, the optimal value being reached after approximately 1 000 iterations. For the function that evaluates the GCI, the algorithm has modified the value of the function even after 9 000 iterations (from a total of 10 000). The optimal values are in the same order of magnitude as in the case of the Particle Swarm Optimization and the Genetic Algorithm and it may be observed that a higher number of variables in the design vector leads to better values of the cost function.

In the Fig. 12 and the Fig. 13 the optimal values (correspondent to the iteration 10 000) of each algorithm are presented, for both of the objective functions are presented on two charts, for the cases with 2, 4 and 6 variables in the design vector.

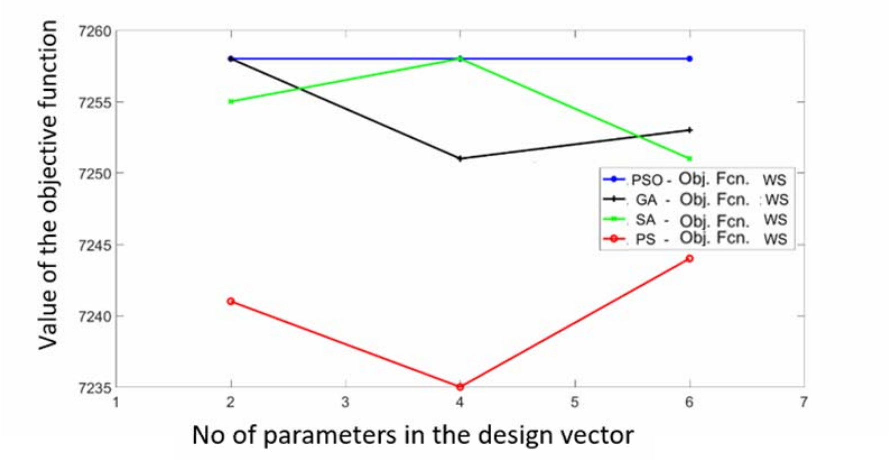

Fig. 12 – Comparative results of the four algorithms for the workspace.

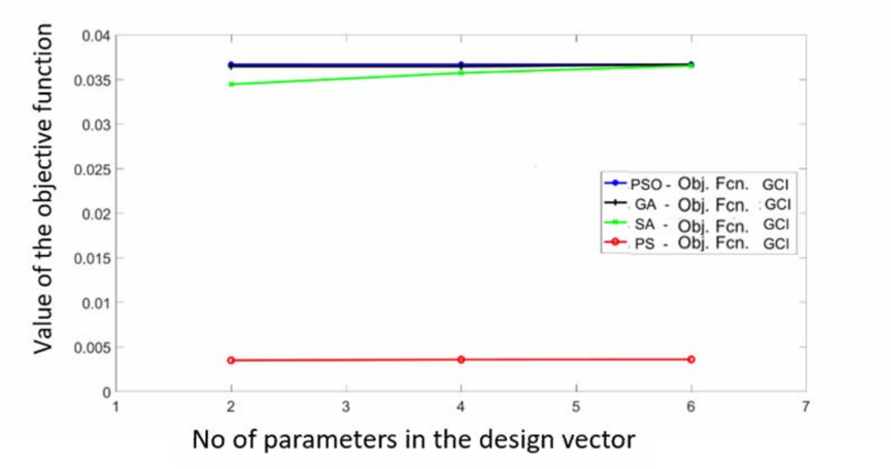

Fig. 13 – Comparative results of the four algorithms for the Global Conditioning Index.

Also, in the Table 1 are presented the optimal value of each algorithm and the necessary computational time correspondent to the best value (on the line labeled with Max). Also, for each algorithm and objective function, it has been evaluated the mean optimal value and the mean computational time after running the algorithm 10 times (presented on the line labeled with Med).

Table 1

Comparative results of the algorithms for the two objective functions

			Value of the cost function and the required computational time			
Alg.	No. Var.	Val.	Workspace value	Computational time [s]	GCI	Computational time [s]
GA	2	Max	7258	1985	0.036442	858
		Med	7246.2	1966.4	0.036421	855.2
	4	Max	7251	1964	0.036434	902
		Med	7247.9	1973.3	0.036425	903.2
	6	Max	7253	1913	0.036654	881
		Med	7251.3	1971.8	0.036649	904.9
PSO	2	Max	7258	1863	0.036673	861
		Med	7257.4	1868.2	0.036668	859.3
	4	Max	7258	1897	0.036673	859
		Med	7258	1889.5	0.036671	862.1
	6	Max	7258	1884	0.036673	864
		Med	7258	1893.4	0.036673	863.2
PS	2	Max	7241	3036	0.003493	3455
		Med	7219.3	3029.2	0.003491	3452.1
	4	Max	7235	3003	0.003574	3462
		Med	7226.2	3031.8	0.003563	3459.3
	6	Max	7244	3105	0.003590	3433
		Med	7239	3095.4	0.003582	3455.2
SA	2	Max	7255	2799	0.034448	2434
		Med	7252.5	2771.5	0.034331	2383.3
	4	Max	7258	2930	0.035718	2582
		Med	7255.8	2856.4	0.035425	2501.8
	6	Max	7251	2852	0.036528	2560
		Med	7256.1	2869.2	0.035991	2544.6

By analyzing the table above, is can be stated the Particle Swarm optimization has reached the maximal absolute value and the maximal mean value of the objective function. These characteristics has been preserved regardless of the number of variables in the design vector and for both of the objective functions. This is why, the Particle Swarm Optimization may be considered as the most efficacious algorithm among all the four algorithms.

On the other side, the Genetic Algorithm and the Simulated Annealing presented high performances, almost comparable to the Particle Swarm Optimization (both for the maximal values as for the mean ones). What is more, these two algorithms have performed equally to the Particle Swarm Optimization in two cases: optimization with the workspace as cost function and two design variables for the Genetic Algorithm and 4 variables for the Simulated Annealing.

The least efficacious algorithm has been, for each case, the Pattern Search. Even though this algorithm presents a fast convergence, it has demonstrated lower performances than the other three algorithms. In the case when the objective function has been the GCI, the algorithm has obtained the results (both for maximal and mean values) with a degree of magnitude lower compared to the others.

Regarding the efficiency, a first aspect that may be observed is the fact that the required computational time varies with the type of the objective function. For evaluating the efficiency, the mean value of the computational time is analyzed. Therefore, in the case when the objective function has been the workspace, the Particle Swarm Optimization has obtained the best values of the fitness function by using the lowest computational times, regardless of the number of variables in the design vector. This is why, the Particle Swarm Optimization is considered to be the most efficient algorithm. Also, the Genetic Algorithm demanded a computational time almost similar to the Particle Swarm Optimization. The algorithm that required the higher computational time is the Pattern Search, being considered the least efficient algorithm.

In the case when the objective function has been the GCI, the lowest mean time has been obtained by the Genetic Algorithm, for the case with 2 variables in the design vector. Still, for the cases with four and six variables, the lowest time corresponds to the Particle Swarm Optimization. On the other hand, the Simulated Annealing and the pattern Search have required computational time up to four times higher as in the case of the Genetic Algorithm or the Particle Swarm Optimization. The least effective algorithm may be considered, again, the Pattern Search.

As a general observation, for each algorithm and cost function, the required computation time of each algorithm is higher with the increase of the number of variables in the design vector but, on the other hand, the returned value of the objective function is improved.

7. CONCLUSIONS

In the present paper, a comparative analysis of the performance of four optimization algorithms (the Genetic Algorithm, the particle Swarm Optimization, the Pattern Search and the Simulated Annealing), frequently used for optimizing the robots has been compared.

The performance of these algorithms has been analyzed on what regard the geometric optimization of a robot, with different number of variables in the design vector and by using two objective functions: the maximization of the workspace and the maximization of the Global Conditioning Index.

Therefore, it may be concluded that, on what regards the efficacy (the ability of an algorithm to reach an optimal value) the most performant algorithm has been the Particle Swarm Optimization, regardless of the number of the variables in the design vector or of the type of the cost function. The least efficient algorithm is the Pattern Search, even though it presents a fast convergence from the first iterations. Regarding the Genetic Algorithm and the Simulated Annealing, these algorithms have performed similarly, their performance being almost equal to the Particle Swarm Optimization.

On what concerns the efficiency, the ability to reach an optimum using a computational time as low as possible, the least mean computational time has been obtained again by the Particle Swarm Optimization.

Regarding the convergence, the algorithms that presents the fastest convergence is the Pattern Search, reaching the optimal value after 500 iterations (5% from the total of 10 000). On the other hand, the Genetic Algorithm and the Particle Swarm Optimization reach an optimal value after 3000 iterations (meaning 30% from the total number). The convergence of the Simulated Annealing has not been constant, the algorithm reaching an optimal value also in the first 500 iterations (5% from the total) but also after 9 000 iterations (90% of the total).

By referring to the current state of the art, it can be concluded that, in the case of optimizing a robot, the most efficient and effective algorithm is the Particle Swarm Optimization, followed by the Simulated Annealing. On the other hand, when a feasible solution is to be identified in a short period of time, the most suitable algorithm is the Pattern Search because it presents a fast convergence (in about 300 iterations), but it reaches a local optimum. This algorithm may be used in the pre-design phase or for a rapid estimation of the real dimensions of a robot.

As a future outlook, a potential development of the work presented in this paper is the comparison of the performance of the optimization algorithms in the case of a multi-objective optimization problem.

Acknowledgements. The authors acknowledge similar and equal contributions to this paper. The authors gratefully acknowledge the financial support received from Romanian Ministry of Research and Innovation, project PN-IIIP2-2.1-PED-2019-0085 CONTRACT 447PED/2020 (Acronym POSEIDON)

Received on July 1, 2021

REFERENCES

1. IQBAL, Kamran, *Fundamental engineering optimization methods*, Londres: Bookboon, pp. 35–50, 2013.
2. NOCEDAL, Jorge, WRIGHT, Stephen, *Numerical optimization*, Springer Science & Business Media, 2006.
3. LUENBERGER, David G., *Linear and nonlinear programming*, Reading, MA, Addison-wesley, 1984.
4. MARLER, R. Timothy, ARORA, Jasbir S., *Survey of multi-objective optimization methods for engineering*, Structural and multidisciplinary optimization, pp. 369–395, 2004.
5. CARAMIA, Massimiliano, DELL'OLMO, Paolo, *Multi-objective management in freight logistics: Increasing capacity, service level and safety with optimization algorithms*, Springer Science & Business Media, 2008.
6. BRIȘAN, Cornel, BOANTĂ, Cătălin, CHIROIU, Veturia, *Introduction in Optimization of industrial robots. Theory and applications*, Editura Academiei Române, 2019.

7. CHAUDHURY, Arkadeep Narayan, GHOSAL, Ashitava, *Optimum design of multi-degree-of-freedom closed-loop mechanisms and parallel manipulators for a prescribed workspace using Monte Carlo method*, Mechanism and Machine Theory, **118**, pp. 115–138, 2017.
8. FU, Jianxun, GAO, Feng, *Optimal design of a 3-leg 6-DOF parallel manipulator for a specific workspace*, Chinese Journal of Mechanical Engineering, **29**, *4*, pp. 659–668, 2016.
9. HAMIDA, Ben, et.al., *Comparative study of design of a 3-DOF translational parallel manipulator with prescribed workspace,* IFToMM World Congress on Mechanism and Machine Science, Springer, Cham, 2019, pp. 501–512.
10. LARIBI, Med Amine,et.al., *Robust Optimization of the RAF Parallel Robot for a Prescribed Workspace*, Computational Kinematics, Springer, Cham, 2018, pp. 383–393.
11. LIU, Xin-Jun, WANG, Jingsong, OH, Kun-Ku, KIM, Jongwon, *A New Approach to the Design of a DELTA Robot with a Desired Workspace*, Journal of Intelligent and Robotic Systems, **39**, *2*, pp. 209–225, 2004.
12. WU, Guanglei, BAI, Shaoping, HJØRNET, Preben, *Architecture optimization of a parallel Schönflies-motion robot for pick-and-place applications in a predefined workspace,* Mechanism and Machine Theory, **106**, pp. 148–165, 2016.
13. MERLET, Jean-Pierre, *Parallel robots*, Springer Science & Business Media, 2006.
14. SICILIANO, Bruno, KHATIB, Oussama (eds.), *Springer handbook of robotics*, Springer, 2016.
15. ZARGARBASHI, Seyedhossein., KHAN, Waseem, ANGELES, Jorge, *The Jacobian condition number as a dexterity index in 6R machining robots*, Robotics and Computer-Integrated Manufacturing, **28(6)**, pp. 694–699, 2012.
16. YOSHIKAWA, Tsuneo, *Manipulability of robotic mechanisms,* The international journal of Robotics Research, **4**, *2*, pp. 3–9, 1985.
17. PATEL, Sarosh, SOBH, T., *Manipulator performance measures-a comprehensive literature survey,* Journal of Intelligent & Robotic Systems, **77**, *3*, pp. 547–570, 2015.
18. GOSSELIN, Clement, ANGELES, Jorge, *A global performance index for the kinematic optimization of robotic manipulators*, pp. 220–226, 1991.
19. LUM, Mitchell JH, et al. *Kinematic optimization of a spherical mechanism for a minimally invasive surgical robot,* IEEE International Conference on Robotics and Automation, Proceedings of ICRA'04, pp. 829–834, IEEE, 2004.
20. MONSARAT, Bruno, GOSSELIN, Clément M., *Workspace analysis and optimal design of a 3-leg 6-DOF parallel platform mechanism*, IEEE Transactions on Robotics and Automation, **19**, *6*, pp. 954–966, 2003.
21. WAN, Yuehua, et al., *A survey on the parallel robot optimization*, Proceedings of II-nd International Symposium on Intelligent Information Technology Application, pp. 655–659 IEEE, 2008.
22. ZHANG, Xiaoli, NELSON, Carl A., *Multiple-criteria kinematic optimization for the design of spherical serial mechanisms using genetic algorithms*, Journal of Mechanical Design, **133**, *1*, 2011.
23. LI, Zhibin, et al., *Type synthesis, kinematic analysis, and optimal design of a novel class of Schönflies-Motion parallel manipulators*, IEEE Transactions on Automation Science and Engineering, **10**, *3*, pp. 674–686, 2012.
24. GAO, Zhen, ZHANG, Dan, *Performance analysis, mapping, and multiobjective optimization of a hybrid robotic machine tool*, IEEE Transactions on industrial electronics, **62**, *1*, pp. 423–433, 2014.
25. LIAO, Bin, LOU, Yunjiang, *Optimal kinematic design of a new 3-DOF planar parallel manipulator for pick-and-place applications*, 2012 IEEE International Conference on Mechatronics and Automation, pp. 892–897, IEEE, 2012.
26. XU, Qingsong, LI, Yangmin, *Stiffness optimization of a 3-dof parallel kinematic machine using particle swarm optimization,* 2006 IEEE International Conference on Robotics and Biomimetics, pp. 1169–1174, IEEE, 2006.

27. ZHANG, Dan, WEI, Bin, *Kinematic analysis and optimization for 4PUS-RPU mechanism*, 2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), pp. 330–335, IEEE, 2015.
28. ZHANG, Zhenchuan, YU, Hongjian, DU, Zhijiang, *Design and kinematic analysis of a parallel robot with Remote Center of Motion for Minimally Invasive Surgery*, 2015 IEEE International Conference on Mechatronics and Automation (ICMA), pp. 698–703, IEEE, 2015.
29. BAI, Qinghai, *Analysis of particle swarm optimization algorithm,* Computer and information science, **3**, *1*, pp. 180–184, 2010.
30. SHI, Yuhui, EBERHART, Russell C., *Empirical study of particle swarm optimization*, Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), IEEE, pp. 1945–1950, 1999.
31. AL-SUMAIT, J. S., AL-OTHMAN, A. K., SYKULSKI, J. K., *Application of pattern search method to power system valve-point economic load dispatch,* International Journal of Electrical Power & Energy Systems, **29**, *10*, pp. 720–730, 2007.
32. ZHANG, Dong, YUN, Chao, SONG, Dezheng, *Dexterous space optimization for robotic belt grinding*, Procedia Engineering, **15**, pp. 2762–2766, 2011.
33. ENFERADI, Javad, NIKROOZ, Reza, *The performance indices optimization of a symmetrical fully spherical parallel mechanism for dimensional synthesis*, Journal of Intelligent & Robotic Systems, **90**, *3-4*, pp. 305–321, 2018.
34. LI, Zixiang, TANG, Qiuhua, ZHANG, LiPing, *Minimizing energy consumption and cycle time in two-sided robotic assembly line systems using restarted simulated annealing algorithm*, Journal of Cleaner Production, **135**, pp. 508–522, 2016.
35. LANNI, C., SARAMAGO, S. F. P., CECCARELLI, M., *Optimal design of 3R manipulators by using classical techniques and simulated annealin,* Journal of the Brazilian Society of Mechanical Sciences, **24**, *4*, pp. 293–301, 2002.
36. ABOULISSANE, Badreddine, EL BAKKALI, Larbi, EL BAHAOUI, Jalal, *Workspace analysis and optimization of the parallel robots based on computer-aided design approach*, Facta Universitatis, Series: Mechanical Engineering, **18**, *1*, pp. 79–89, 2020.
37. AYYILDIZ, Mustafa, ÇETINKAYA, Kerim, *Comparison of four different heuristic optimization algorithms for the inverse kinematics solution of a real 4-DOF serial robot manipulator*, Neural Computing and Applications, **27**, *4*, pp. 825–836, 2016.
38. ERDOGMUS, Pakize, TOZ, Metin, *Heuristic optimization algorithms in robotics*, Serial and Parallel Robot Manipulators-Kinematics, Dynamics, Control and Optimization, pp. 311–338, InTech Publisher, 2012.
39. GUPTA, Surbhi, SARKAR, Sankho Turjo, KUMAR, Amod, *Design optimization of minimally invasive surgical robot*, Applied soft computing, **32**, pp. 241–249, 2015.
40. LOU, Yunjiang, et al., *Optimization algorithms for kinematically optimal design of parallel manipulators*, IEEE Transactions on Automation Science and Engineering, **11**, *2*, pp. 574–584, 2013.
41. SAVIN, Sergei, *Parameter Optimization for Walking Patterns and the Geometry of In-Pipe Robots*, 2018 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), pp. 1–6, IEEE, 2018.
42. ENFERADI, Javad, NIKROOZ, Reza, *The performance indices optimization of a symmetrical fully spherical parallel mechanism for dimensional synthesis*, Journal of Intelligent & Robotic Systems, **90**, *3-4*, pp. 305–321, 2018.
43. ZHANG, Dan. *Parallel robotic machine tools*, Springer Science & Business Media, New York, pp. 93–115, 2009.
44. PLITEA, N., HESSELBACH, J., PISLA, D., RAATZ, A., VAIDA, C., WREGE, J., BURISCH, A.: *Innovative development of parallel robots and microrobots*, Acta Tehnica Napocensis, Series of Applied Mathematics and Mecanics 5 ,**49**, pp. 5–26, 2006.
45. HUSTY, M., BIRLESCU, I., TUCAN, P., VAIDA, C., PISLA, D.: *An algebraic parameterization approach for parallel robots analysis,* Mechanism and Machine Theory, **140**, pp. 245–257, 2019.