ISSN 2176-5480

22nd International Congress of Mechanical Engineering (COBEM 2013)
November 3-7, 2013, Ribeirão Preto, SP, Brazil

OPTIMUM DESIGN OF 3R ROBOT MANIPULATOR BY USING IMPROVED DIFFERENTIAL EVOLUTION IMPLEMENTED IN PARALLEL COMPUTATION

Milena Almeida Leite Brandão
Sezimária de Fátima Pereira Saramago
Federal University of Uberlândia, Ituiutaba - MG - Brazil
Federal University of Uberlândia, Uberlândia - MG - Brazil
milena@pontal.ufu.br, saramago@ufu.br

José Laércio Doricio
Federal University of Uberlândia, Uberlândia - MG - Brazil
jldoricio@pontal.ufu.br

Abstract. *In recent decades the great interest in Evolutionary Algorithms (EAs) has boosted their development leading to a significant improvement in their efficiency and applicability. Thus, EAs have been applied to solve optimization problems in different areas of knowledge. A promising optimization method known as Differential Evolution (DE), which belongs to the class of AEs, has attracted the attention of researchers. The DE algorithm is simple, robust and efficient. However, by testing with classical optimization problems noticed that sometimes the results obtained with DE are not as satisfactory as expected or that in many cases the algorithm ends the search for the optimal solution prematurely. Recently, with the advancement and greater availability of computer technology, the scientific community has been thinking about the implementation of optimization algorithms in parallel in order to reduce the processing time. The main objective of this paper is to present an improvement of the Differential Evolution optimization method, proposing modifications to the basic algorithm by using shuffled complex and making it able to work with parallel computing. The proposed methodology is applied to the optimal design of an orthogonal 3R robot manipulator that takes into account the characteristics of its workspace. For this purpose, a multi-objective optimization problem is formulated to obtain the optimal geometric parameters for the robot. The maximum workspace volume, the maximum system stiffness and the optimum dexterity are considered as the multi-objective functions. The results show that the procedure represents a promising alternative for the type of problem presented above.*

Keywords: *Optimization, Robotics, Differential Evolution, Parallel Computation, Improved Differential Evolution*

1. INTRODUCTION

Optimization is an important tool for decision-making during the analysis and design of physical systems. After the formulation of the model, an optimization algorithm is used to obtain the solution of the problem, usually with the help of computer codes. There is no universal algorithm, but a set of methods, some of which are more suitable for specific applications. The choice of the optimization method depends on the user and can determine the effectiveness or failure to solve the problem.

The Evolutionary Algorithms (EAs) are general methods for stochastic search optimization which have the theory of evolution as their basis for formulation. In recent decades, the great interest in these algorithms has driven their development which has led to a significant improvement in the efficiency and applicability of EAs to solve optimization problems in different areas of knowledge.

An algorithm belonging to the class of AEs that has emerged is the optimization method known as Differential Evolution (DE). The DE algorithm is simple, easy, robust and efficient. But since it is a relatively new technique, emerged in the mid-90s, many researchers have proposed modifications to the original algorithm of DE aiming to improve its convergence. By testing with classical optimization problems, we noticed that sometimes the results obtained with the DE are not as good as expected. Furthermore, in many cases the algorithm terminates the search for the optimal solution prematurely.

The DE algorithm starts by creating an initial population of N_p individuals, randomly chosen, which should cover the entire search space. It is usually created by a uniform probability distribution, when there is no knowledge about the problem.

Each individual, called vector, has n components represented by real values that are the number of design variables. Thus, a natural population shows that the number of individuals is generally constant for all generations.

The main idea of differential evolution is to generate new individuals, denoted vectors modified or donors, by adding

ISSN 2176-5480

the weighted difference between two random individuals from the population to a third individual. This operation is called mutation.

The components of the individual donor are mixed with the components of a randomly chosen individual denoted target vector, to yield the so-called trial vector, or experimental vector. This process is referred to as crossover. If the experimental vector result in an objective function value is less than the target vector, then the trial vector replaces the target vector in the next generation. This operation is called selection. The procedure is terminated when a stopping criterion is reached.

Although several positives aspects, it has been observed that DE sometimes does not have as good a performance as expected. The empirical analysis of DE has shown that the algorithm may fail to proceed towards a global optimum tending to a state of stagnation, in which individuals are very similar to each other, that is, the population is homogeneous and the algorithm shows no improvement despite accepting new individuals in the population. In addition, DE also suffers from the problem of premature convergence. This situation arises when there is a loss of population diversity. As a result, the population converges to a point which can not be a global optimum solution. This usually occurs when the objective function is multimodal.

As in other evolutionary algorithms, the performance of DE decreases with the increase of the size of the objective function. Many other authors have suggested several modifications in the structure of this algorithm to improve its performance.

Raupp *et al.* (2010) presented a new metaheuristic based on Differential Evolution for the general problem of nonlinear programming called Perfected Differential Evolution (PDA). Counting on innovations in the use of crossover, mutation and selection operators, PDA works starting from a reformulation of the original problem into of a box-constrained bi-objective optimization problem. Additionally, PDA introduces an additional stopping criterion that allows the detection of convergence of the population, which avoids unnecessary computational operations and, consequently, increases its efficiency.

In the article of Das *et al.* (2005), they proposed two modifications to the basic scheme of Differential Evolution: (1) the introduction of the concept of scale factor by which time-varying vector difference will be multiplied, and (2) the variation of the scale factor in a random mode. The goal of these modifications is to try to prevent or slow the premature convergence in the early stages of the search and facilitate convergence to a global optimal solution during later stages of the research.

According to Li *et al.* (2011), in Differential Evolution algorithm the vector mutation is generated starting from a suboptimal vector and two other individuals in the population. The purpose of this change is to accelerate the convergence speed of the method without the occurrence of a premature convergence.

Other suggestions for modification of the algorithm of DE can be found in the articles: Coelho *et al.* (2009), Mehablia (2010) and Chong *et al.* (2012). These are just some examples of the academic community's efforts to improve the global convergence of Differential Evolution. Many proposed changes involve the same principle, to change some aspect of the mutation, crossover and selection operations of DE.

Although EAs are algorithms that can solve a wide variety of optimization problems, real problems of design optimization require much computational effort and the response of the algorithm can take weeks or months. With the advancement and availability of computer technology, the scientific community has considered the implementation of optimization algorithms in parallel in order to shorten the processing time.

In work shown by Kwedlo and Bandurski (2006), they proposed a new parallelization scheme for the calculation of the value of the objective function. This scheme is based on the decomposition of data; both the learning set and the population of the evolutionary algorithm are distributed among processors. The processors form a pipeline using the ring topology. In a single step each processor computes the objective function value of a subpopulation while sending the previous subpopulation to the subsequent processor and receives the next subpopulation from the preceding processor. According to the authors, in this way it is possible to overlap communication and computation, and the first experimental results show that, for large datasets, the algorithm obtains very good, almost linear, speedup values.

In the paper of Tasoulis *et al.* (2004) DE is parallelized on a parallel virtual environment in order to improve the speed and performance of the method. The authors report experimental results indicating that the exchange of information between subpopulations assigned to different processors has a significant impact on the algorithm performance.

Bujok (2011) argues that the main optimization problem is the time and complexity to find the global optimum and a way to solve this problem is the use of parallel models of the DE algorithm. Furthermore, according Bujok (2011), there are several approaches in parallelism that can lead to a faster search for a global minimum. The parallel models most used are the master-slave model, the island or migration model, the diffusion model and the combination of these results in hybrid models. In his article, the migration model is used to parallelize DE as follows: the population is divided into several sub-populations (islands) and individuals migrate between them.

In this study, we propose a modification to the basic layout of the DE, adding to its algorithm the concept of evolution shuffled sets, in which the central idea is to divide the population into multiple sets or subpopulations and let each set evolve separately. This technique increases the diversity of the population helping to avoid premature convergence. After

ISSN 2176-5480

evolving, the sets are grouped to form a new population, which is then shuffled and divided again into subpopulations. This procedure continues until it satisfies some stopping criterion.

Furthermore, since the algorithm showed a highly parallelizable aspect, since each set can evolve in different processors, the modified DE algorithm is implemented in parallel with the objective of reducing the processing time during program execution, for their application in complex problems.

Industrial robots and computer-aided systems are the latest trend in fabrication process automation, since the advances in the sensors field allow the development of more sophisticated tasks. The use of robots in the industry is wide since they accomplish tasks that are dangerous or monotonous for humans, according to Vargas *et al.* (1992). This is the case of an industrial robot used for cleaning in electrical substations, which are high voltage areas. According to Bergamaschi *et al.* (2006) and Bergamaschi *et al.* (2008) in robotic manipulators, a fundamental characteristic that must be taken into account in the dimensional design is the volume of their workspace. It is crucial to calculate the workspace and its boundaries with perfect precision, because they influence the dimensional design, the manipulator's positioning in the work environment, and its dexterity to execute tasks.

In literature, several investigations have focused on the properties of the workspace of open chain robotics with the purpose of emphasizing its geometric and kinematic characteristics. Ceccarelli (1996) presented an algebraic formulation to determine the workspace of revolution manipulators. Lanni *et al.* (2002) investigated and solved the design of manipulators in the form of an optimization problem that takes into account the characteristics of the workspace. They applied two different numerical techniques: the first using sequential quadratic programming (SQP) and simulated annealing. Abdel-Malek *et al.* (2000) proposed a generic formulation to determine voids in the workspace of serial manipulators. Other researches have focused on determining the workspace boundary and on detecting the presence of voids and singularities in the workspace. Saramago *et al.* (2002) proposed a form of characterizing the workspace boundary, formulating a general analytic condition to deduce the existence of cusp points in the interior and exterior boundaries of the workspace. Ceccarelli and Lanni (2004) presented a suitable formulation for the workspace that can be used in the design of manipulators, which was formulated as a multi-objective optimization problem using the workspace volume and robot dimensions as objective functions. Bergamaschi *et al.* (2006) and Bergamaschi *et al.* (2008) studied the design of manipulators with three-revolute joints (3R) using an optimization problem that takes into account the characteristics of the workspace. The optimization problem is formulated considering the workspace volume as the objective function. Constraints are added to guarantee the regularity of the envelope and force the workspace to occupy a pre-established area.

In this paper, the developed methodology is applied to the optimal design of a 3-revolute (3R) robot manipulator with orthogonal axes taking into account the characteristics of its workspace. For this purpose, a multi-objective optimization problem is formulated to obtaining the optimal geometric parameters for the robot. The maximum workspace volume, the maximum system stiffness and the optimum dexterity are considered as the multi-objective functions. This work is organized as follows. A review dedicated to the DE technique is presented in Section 2. Section 3 present a brief review about the multi-objective problems. Section 4 present the general aspects regarding the mathematical modeling of the manipulator. The results and discussion are presented in Section 5. Finally, the conclusions are outlined in Section 6.

2. Improved Differential Evolution

The Shuffled Complex Evolution algorithm, SCE, (Duan *et al.* (1993)) deals with the search for the global optimum as a process of natural evolution. The N_p sampling points constitute a population which is divided into multiple sets, or subsets, such that each set has the same number of individuals. Each of the sets is allowed to evolve independently, or exploring the search space in different directions. After a certain number of generations, the sets are grouped together and mixed again, so new sets are obtained by a shuffling process.

An algorithm for the SCE optimization method could be seen in the work of Duan *et al.* (1993) and in some applications in Brandão *et al.* (2011).

The basic algorithm of Differential Evolution modified using the concept of evolution with scrambled sets is called Improved Differential Evolution, or IDE. The IDE starts as a usual DE algorithm for creating a population of individuals randomly sampled from the feasible region using uniform probability distribution. The population is then sorted in ascending order of values of the objective function and partitioned into multiple sets. Each set executes the DE independently. In the evolution stage, the sets are forced to mix and the points are reassigned to ensure the exchange of information. The process of the proposed IDE algorithm is described in the flowchart shown in Fig. 1.

The idea of dividing program tasks across multiple processors is old but has only recently become feasible due to rapid advancement of hardware and software. Machines have evolved greatly in speed and multiprocessor capability, and programs that utilize these resources are showing a great increase in efficiency. Thus, the objective is to reduce the processing time during the execution of the improved algorithm to enable its application in complex problems.

In the master processor, an initial population is randomly created which is divided into k subsets ($k \in \mathbb{Z}$ is less than or equal to the number of processors) and distributed among a maximum of k processors. Then the DE code continues processing sequentially on each processor until some stopping criterion is reached. Subpopulations are then grouped in the master processor, scrambled and divided again into subpopulations. The flowchart of the Improved Differential Evolution

ISSN 2176-5480

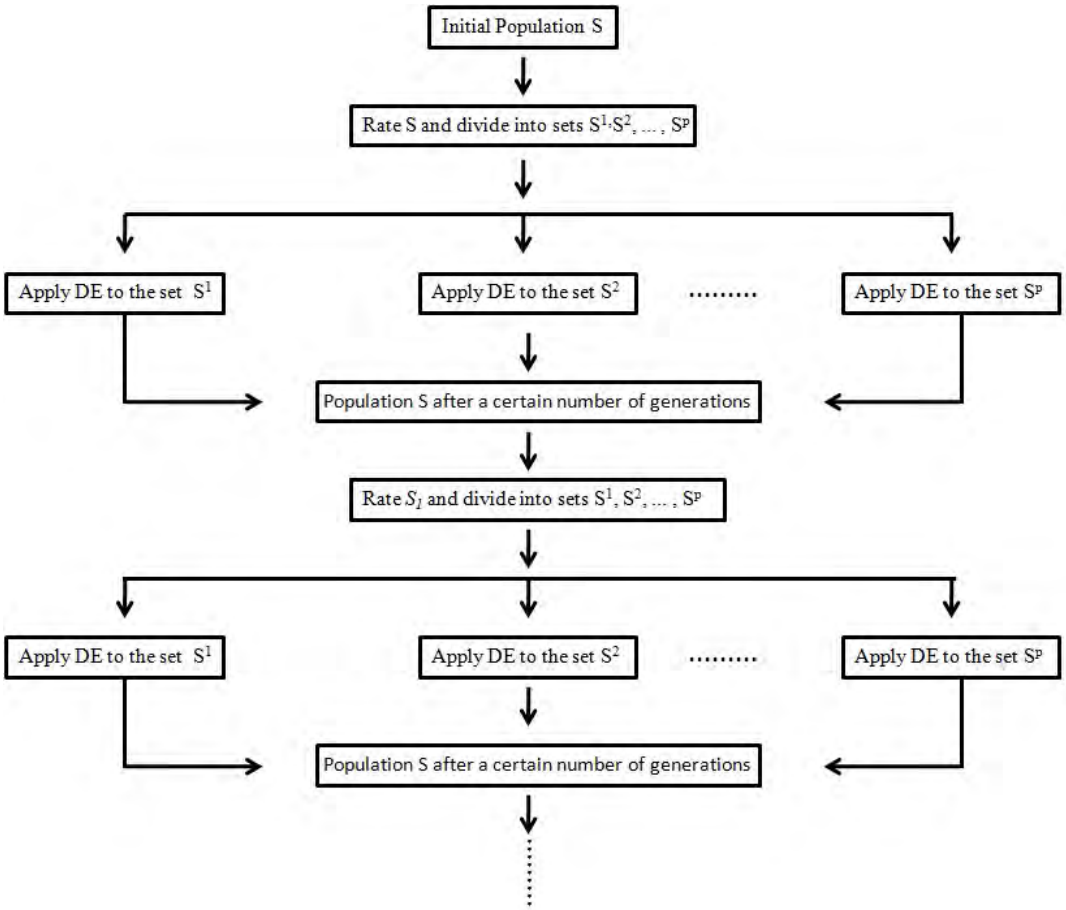


Figure 1: Flowchart of Improved Differential Evolution Method

Optimization Method implemented in parallel can be seen in Fig. 2.

Then the IDE method implemented in parallel (IDEP) will be used to obtain the optimal design of 3R orthogonal manipulators through modeling developed by Oliveira and Saramago (2010).

3. Multiobjective Optimization

Several problems of engineering and other areas have multiple objectives to be achieved and in these cases the mathematical models are no longer represented by a single function, but rather by various objective functions simultaneously. These problems are known as multi-objective optimization problems.

Intuition might lead one to believe that in order to maximize or minimize a multiobjective problem, it would suffice to minimize or maximize each function separately. But the optimal solution of a function is not always the optimal point for the other functions. Therefore, the expected solution to multiobjective optimization consists of a set of feasible solutions.

Thus, modeling in multiobjective optimization is formulated so as to find a vector of design variables, which is represented by a column vector of n variables $x = [x_1 x_2 \cdots x_n]^T \in \mathbb{R}^n$, which satisfies both the constraints of the problem and optimizes a vector of objective functions.

There are many methods which can be used to minimize a multiobjective function by obtaining the Pareto optimal solution or a set of such solutions. Once you've found this set the user choose the best solution based on the information we already have about the multi-objective function and its constraints. Some of these methods are shown below.

In the methods that will be studied in this work, the set of functions is replaced by a single scalar function that represents all the objective functions. Thus, obtaining the optimum solution is reduced to minimize a single function. The analysis of the solution vector is accomplished by modifying the main function according to the importance of each goal. Therefore a set of solutions is selected according to the changes made in the main function, and it is up to the researcher to analyze them to choose the best solution among them.

ISSN 2176-5480

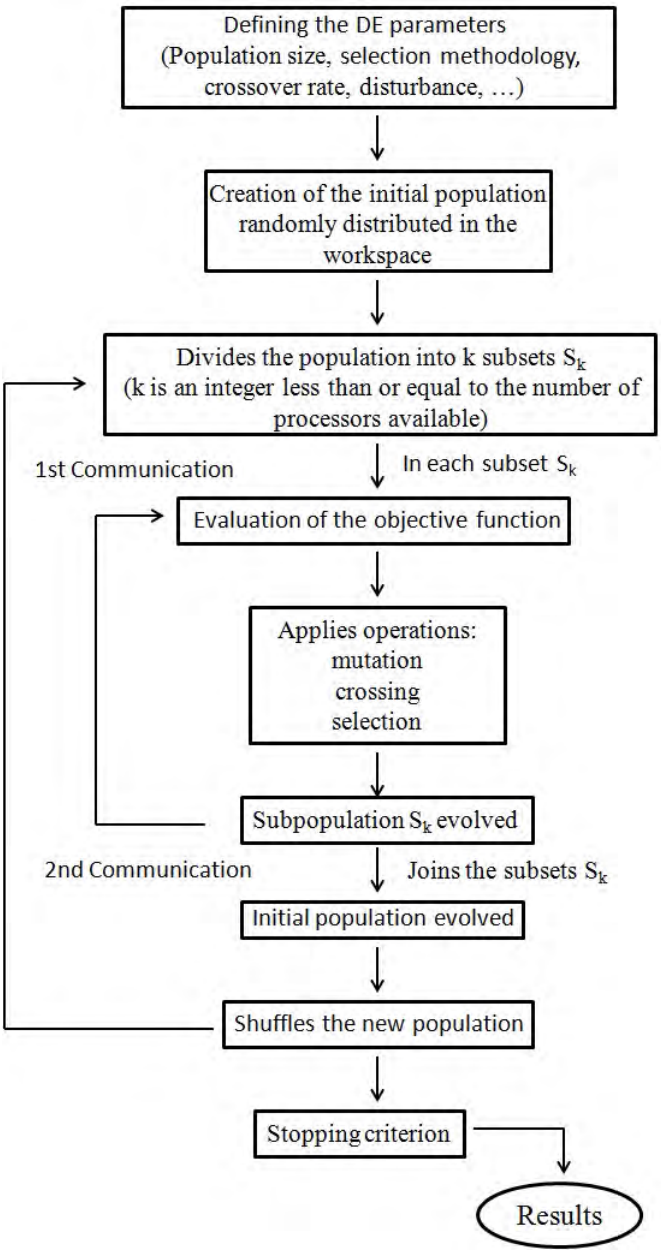


Figure 2: Flowchart of Improved Differential Evolution Algorithm with Parallel Processing

3.1 Method of Weighting of Objectives

In this method, the multi objective optimization problems are reformulated and replaced by a single scalar optimization problem. The main idea is to create a single primary function from the objective functions, giving them a degree of importance, that is, considering their values according to their importance.

This manipulation of algebraic functions is given in the form:

$$f(x) = \sum_{i=1}^{n} f_i(x) \tag{1}$$

where $w_i \geq 0$ are the weights representing the relative importance of each criterion. It must be assumed that:

$$\sum_{i=1}^{n} w_i = 1 \tag{2}$$

Taking into account that the results obtained in solving a problem using the modeling described in Eq.(1) can vary significantly with the changing values of the weightings and that still little is known about how to choose these coefficients, it becomes necessary to obtain different approaches varying the values of w_i to solve the same problem. Once obtained,

ISSN 2176-5480

they are compared to choose the best among them. This choice is made by means of the user experience and knowledge of the problem.

Note that the search for the optimal point using Eq.(1) does not depend only on the values of w_i but also on the units in which the functions are expressed.

Thus, for w_i to express the importance of all the objective functions, they must all be expressed in dimensionless form. This study will use the optimal solution values of each function f_i^0 as follows:

$$f(x) = \sum_{i=1}^{n} w_i f_i(x) c_i \tag{3}$$

c_i being constant multipliers.

The method of weighting objectives is most appropriate when you want to prioritize an objective and for this reason the result is strongly influenced by the choice of the most important objective.

3.2 Global Criteria Method

Global Criteria method uses a value established as ideal for each function as a basis of calculation to define the degree of importance of each point x of the feasible region of parameters. Thus, the optimal solution is a vector of decision variables which minimizes a global criterion. This method converts the multi-objective in a single objective expressed mathematically by the following function:

$$f(x) = \sum_{i=1}^{n} \left(\frac{f_i^0 - f_i(x)}{f_i^0} \right)^s \tag{4}$$

which usually uses $s = 1$ or $s = 2$, but other values can be adopted.

It is clear that for each value of s, a solution obtained by minimizing the equation Eq.(4) will be different. Therefore, the problem is to determine what value of s will result in the solution that is most suitable for the researcher. Also, we can not guarantee the existence of satisfactory solutions because it can happen that for any value of s chosen, the method will provide an unacceptable solution from the point of view of the user.

Equation (4) is not the only way to formulate the global multiobjective optimization criterion function. Another possible way is through a family of metrics L_P defined as:

$$L_p(f) = \left(\sum_{i=1}^{n} \left| f_i^0 - f_i(x) \right|^s \right)^{1/s}, \quad 1 \leq s \leq \infty \tag{5}$$

If $s = 1$ a metric comes down to $L_1(f) = \sum_{i=1}^{n} \left| f_i^0 - f_i(x) \right|$. On the other hand, if s=infinity, we have $L_\infty(f) = max|f_i^0 - f_i(x)|$. The metrics used are over $L_1(f), L_2(f)$ and $L_3(f)$. Note that the minimization of $L_2(f)$ is equivalent to minimizing the Euclidean distance between the function value and the ideal solution.

There is another technique in which instead of working with the distance in an absolute sense, it uses the value of the relative distances. This technique is generally given by:

$$L_p(f) = \left(\sum_{i=1}^{n} \left| \frac{f_i^0 - f_i(x)}{f_i^0} \right|^s \right)^{1/s}, \quad 1 \leq s \leq \infty \tag{6}$$

Because of the many formulations of global criteria, it is important that you apply some of these to make it possible to compare the solutions and then choose the best among them.

Global methods are indicated for cases where you want to get a compromise that seeks to meet all objective functions.

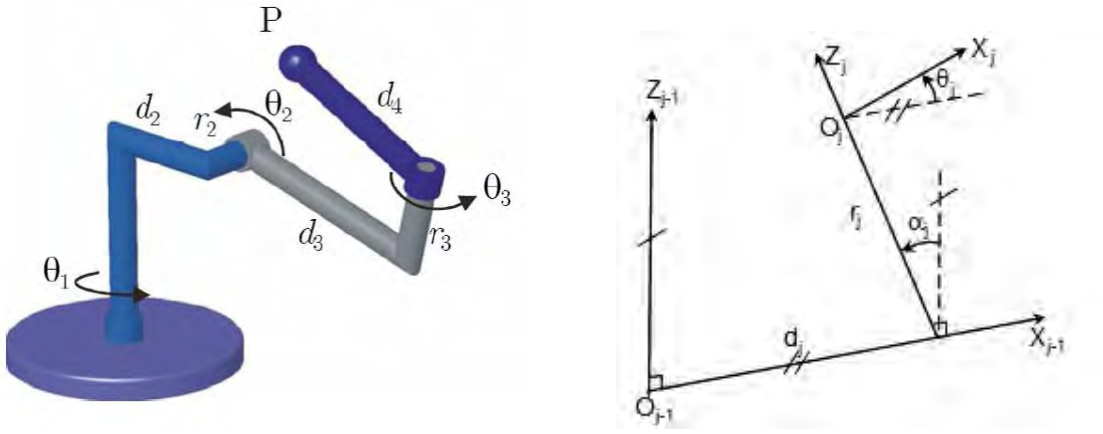
4. Mathematical Modeling of the Robotic Manipulator

Manipulators are multifunctional machines capable of handling parts, tools and other specific devices and perform a variety of tasks. Mathematical models have been created to describe the workspace of these robots and formulate mathematical equations to be used in the graphical representation of this workspace. The manipulators with three rotational joints with orthogonal axes are described in Fig. 3(a).

The study of this type of manipulator is done according to the Denavit-Hartenberg parameters: d_2, d_3, d_4, r_2 and r_3. The joint variables are 1, 2 and 3 which represent the input angles of the actuators.

Considering the axes reference system R_{j-1} and R_j shown in Fig. 3(b), the axes are related by the transformation matrix written as:

ISSN 2176-5480



(a) 3R manipulator with orthogonal axes (b) Denavit-Hartenberg parameters - reference axes

Figure 3: Representation of Denavit-Hartenberg parameters to 3R manipulator

$$T_j^{j-1} = \begin{bmatrix} c_j & -s_j & 0 & d_j \\ c\alpha_j s_j & c\alpha_j c_j & -s\alpha_j & -r_j s\alpha_j \\ s\alpha_j s_j & s\alpha_j c_j & c\alpha_j & r_j c\alpha_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where $c_j = \cos(\theta_j)$, $s_j = \sin(\theta_j)$, $c\alpha_j = \cos(\alpha_j)$ e $s\alpha_j = \sin(\alpha_j)$, $j = 1, \cdots, n+1$.

Thus, the end-effector of a robot with n degrees of freedom is assigned a coordinate system T_n^0 which is a homogeneous transformation matrix.

For this type of manipulator, the direct kinematic model is given by Eqs. (8), (9) and (10):

$$\begin{aligned} x = \; & [d_2 + (r_3 s\alpha_3 - d_4 c\alpha_3 s_3) s_2 + (d_3 + d_4 c_3) c_2] c_1 + \{s\alpha_2 (r_2 + r_3 c\alpha_3 + d_4 s\alpha_3 s_3) + \\ & c\alpha_2 [(r_3 s\alpha_3 - d_4 c\alpha_3 s_3) c_2 - (d_3 + d_4 c_3) s_2]\} s_1 \end{aligned} \tag{8}$$

$$\begin{aligned} y = \; & [d_2 + (r_3 s\alpha_3 - d_4 c\alpha_3 s_3) s_2 + (d_3 + d_4 c_3) c_2] s_1 + \{s\alpha_2 (r_2 + r_3 c\alpha_3 + d_4 s\alpha_3 s_3) + \\ & c\alpha_2 [(r_3 s\alpha_3 - d_4 c\alpha_3 s_3) c_2 - (d_3 + d_4 c_3) s_2]\} c_1 \end{aligned} \tag{9}$$

$$z = c\alpha_2 (r_2 + r_3 c\alpha_3 + d_4 s\alpha_3 s_3) - s\alpha_2 [(r_3 s\alpha_3 - d_4 c\alpha_3 s_3) c_2 - (d_3 + d_4 c_3) s_2] \tag{10}$$

in which $c_i = \cos\theta_i$ and $s_i = \sin\theta_i$, for i = 1, 2, 3.

This research will consider 3R manipulators with orthogonal axes, in other words, $\alpha_2 = -90$ and $\alpha_3 = 90$. As mentioned earlier, a multi-objective optimization problem will be formulated to obtain the optimal parameters for the robot manipulator. The mathematical formulation for calculating the workspace volume, the system stiffness and the robot dexterity will be presented below.

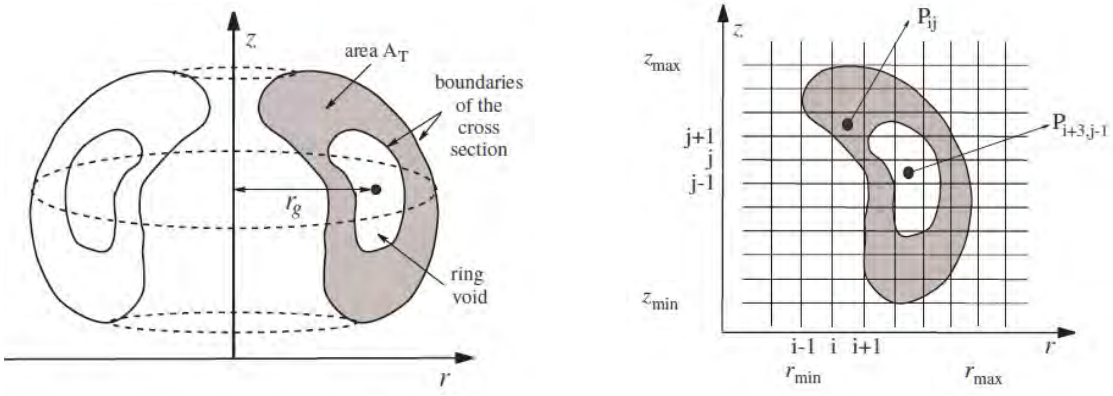
4.1 Workspace of 3R Manipulators

According to Bergamaschi *et al.* (2006), the workspace W is the set of all attainable points for a point P of the end-effector when the joint variables sweep their entire definition interval. Point P is usually chosen as the center of the end-effector, or the tip of a finger, or even the end of the manipulator itself. The first procedure to investigate the workspace is to vary the angles θ_1, θ_2 and θ_3 in their interval of definition and to estimate the coordinates of point P with respect to the manipulator base frame. The workspace of this robot is a solid of revolution. Thus, it is natural to imagine that the workspace is the result of rotation around the z axis of a radial plane section.

The workspace of a three-revolute open chain manipulator can be given in the form of the radial reach r and axial reach z with respect to the base frame, according to Bergamaschi *et al.* (2006). For this representation, r is the radial distance of a generic workspace point from the z-axis, and z is the distance of this same point at the XY-plane (see, Fig. 4(b)). Thus, using Eqs. (8),(9) and (10) the parametric equations (of parameters θ_2 and θ_3) of the geometrical locus described by point P on a radial plane are:

$$r^2 = x^2 + y^2 \tag{11}$$

ISSN 2176-5480



(a) Workspace volume. (b) Discretization of cross section.

Figure 4: A scheme for evaluating the workspace volume of 3R manipulators and discretization of the cross section area by using a rectangular mesh.

where x, y and z are given in Eqs. (8),(9) and (10).

The workspace volume V can be evaluated by the Pappus-Guldin Theorem, using the following equation (see Fig. 4(a)):

$$V = 2r_g A_r \tag{12}$$

where A_r is the cross section area, which is formed by the family of curves given by Eqs. (8),(9) and (10).

This research proposes a numerical formulation to approximate the cross section area, through its discretization within a rectangular mesh. Initially, the extreme values of vectors r and z should be obtained as:

$$r_{min} = \min\, r, \quad r_{max} = \max\, r, \quad z_{min} = \min\, z, \quad z_{max} = \max\, z \tag{13}$$

Adopting n_r and n_z as the number of intervals chosen for the discretization along the r and z axis, the sizes of the elementary areas of the mesh can be calculated:

$$\Delta_r = (r_{\max} - r_{\min})/n_r \tag{14}$$

$$\Delta_z = (z_{\max} - z_{\min})/n_z \tag{15}$$

The n_r and n_z values must be adopted so that the sizes of the elementary areas (Δ_r or Δ_z) are at least 1% of the total distances considered in the discretization ($r_{max} - r_{min}$ or $z_{max} - z_{min}$). Every point of the family of curves form the cross section of the workspace is calculated by Eqs. (8),(9) and (10). Using this equation, by varying the values of θ_2 and θ_3 in the interval $[-\pi, \pi]$, it is possible to obtain the family of curves of the workspace. Given a certain point (r,z), its position inside the discretization mesh is determined through the following index control:

$$i = \text{int}\,[(r - r_{\min})/\Delta r] + 1 \tag{16}$$

$$j = \text{int}\,[(z - z_{\min})/\Delta z] + 1 \tag{17}$$

where i and j are computed as integer numbers. As shown in Fig. 4(b), the point of the mesh that belongs to the workspace is identified by P_{ij}=1, otherwise P_{ij}=0, which means:

$$P_{ij} = 0, \text{ if } P_{ij} \notin W(P) \text{ or } 1, \text{ if } P_{ij} \in W(P) \tag{18}$$

where $W(P)$ indicates workspace region.

In this way, the total area is obtained by the sum of every elementary area of the mesh that is totally or partially contained in the cross section. In Eq. (18), it is observed that only the points that belong to the workspace contribute to the calculation of the area A_T. The coordinate r_g of the center of the mass is calculated considering the sum of the center of the mass of each elementary area, divided by the total area, using the following equation:

$$A_T = \sum_{i=1}^{i_{\max}} \sum_{j=1}^{j_{\max}} (P_{ij}\, \Delta r\, \Delta z) \tag{19}$$

$$r_g = \left(\sum_{i=1}^{i_{\max}} \sum_{j=1}^{j_{\max}} (P_{ij}\, \Delta r\, \Delta z)\, ((i-1)\, \Delta r + (\Delta r/2) + r_{\min}) \right) / A_T \tag{20}$$

Finally, after the calculation of the cross section area and the coordinate of the center of the mass, given by Eqs. (19) and (20), the workspace volume of the manipulator can be evaluated by using Eq. (12).

ISSN 2176-5480

4.2 System Stiffness

From the mechanical viewpoint, stiffness is the measurement of the ability of a body or structure to resist deformation due to the action of external forces. The stiffness of a serial mechanism at a given point of its workspace can be characterized by its stiffness matrix. This matrix relates the forces and torques applied at the gripper link in Cartesian space to the corresponding linear and angular Cartesian displacements.

Two main methods have been used to establish mechanism stiffness models. The first one is called matrix structural analysis, which models structures as a combination of elements and nodes. The second method relies on the calculation of the serial mechanism's Jacobian matrix which is adopted in this work. Matrix J is usually termed Jacobian matrix which is described in Eq. (21). By considering the case that d_2=1, its determinant is calculated by using the Eq. (22).

$$[J] = \begin{pmatrix} -\sin\theta_3\cos\theta_2 d_4 - \cos\theta_2 r_2 & 0 & -\sin\theta_3 d_4 \\ \sin\theta_3\sin\theta_2 d_4 + \sin\theta_2 r_2 & d_3 + \cos\theta_3 d_4 & 0 \\ \cos\theta_2 d_3 + \cos\theta_2\cos\theta_3 d_4 + d_2 & 0 & \cos\theta_3 d_4 \end{pmatrix} \tag{21}$$

$$\det(J) = d_4\,(d_3 + d_4\cos\theta_3)\,[d_2\sin\theta_3 + (d_3\sin\theta_3 + (d_3\sin\theta_3 - r_2\cos\theta_3)\cos\theta_2)] \tag{22}$$

The stiffness matrix of the mechanism in the Cartesian space is then given by the Eq. (23), where K_j is the joint stiffness matrix of the mechanism, with K_j=[k_1,k_2,k_3]. In this case, each actuator of the mechanism is modeled as an elastic component. k_i is a scalar representing the joint stiffness of each actuator, which is modeled as a linear spring:

$$K_C = [J]^T K_j [J] \tag{23}$$

Particularly, in the case where all the actuators have the same stiffness, e.g., $k = k_1 = k_2 = k_3$, Eq. (23) will be reduced to:

$$K_C = k[J]^T [J] \tag{24}$$

Furthermore, the diagonal elements of the stiffness matrix are used as the system stiffness value. These elements represent the pure stiffness in each direction, and they reflect the rigidity of machine tools more clearly and directly. The objective function for system stiffness optimization can be written as Eq. (25). In this case, the stiffness index S can be maximized:

$$S = K_{11} + K_{22} + K_{33} \tag{25}$$

4.3 Dexterity

The condition number of the Jacobian matrix will be used as a measure of dexterity indices for the 3R manipulator. By using the spectral norm, these indices will be described as.

$$Cond(J) = |\lambda_{\max}(J)/\lambda_{\min}(J)| \tag{26}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ mean the maximum and minimum singular values of the Jacobian matrix J, respectively. Regarding the computing time of the optimization process, this expression is selected as the objective function for the optimization of dexterity. The value of $Cond(J)$, which is directly related to singular values of the Jacobian matrix, is between 1 and positive infinity. All the singular values of the Jacobian matrix will be the same and the manipulator is isotropic if $Cond(J)$ is equal to 1. While $Cond(J)$ is prone to be positive infinity it also means that the Jacobian matrix is singular. Therefore, for the optimization of dexterity, the condition number must be minimized.

5. Numerical Simulation

The objective of the proposed manipulator design procedure is the dimensional synthesis of the 3R orthogonal robot. In this study, the optimal design should take into account: the workspace volume (V), the system stiffness (S) and the manipulator dexterity (Cond J). In this way, the multi-objective optimization problem is defined as:

$$\max f(x) = [V \quad -Cond(J) \quad S] \tag{27}$$

subject to $0 \leq x_i \leq 3$, $i = 1, \cdots, 4$.

ISSN 2176-5480

In this formulation, the robot geometric parameters are adopted as the design variables, thus $x = [d_3 d_4 r_2 r_3]^T$. The volume workspace is given by Eq. (12) and the system stiffness is calculated by Eq. (25). Notice that to optimize dexterity the condition number (Cond J), given by Eq. (26), must be minimized.

In Brandão *et al.* (2012), a simplified scheme for the optimal design of the robot manipulator is solved by the Differential Evolution and Shuffled Complex Evolution optimization methods.

In order to evaluate the performance of the DE and IDEP techniques, some important points should be emphasized, as shown below.

According to Oliveira (2006), the Differential Evolution algorithm was performed applying the computer code developed by the authors, implemented in Matlab®, using a PC Intel(R), Core(TM)i7 920 CPU, 2.67 GHz, 3.23 GB (RAM). The parameters adopted for the DE were: population with 15 individuals, 50 generations, ED/best/1/bin strategy, difference factor F = 0.8 and crossover probability CR = 0.6.

The stopping criterion of the algorithm DE, adopted by Oliveira (2006), was the maximum number of population generations and the verification of its stagnation. According to the author, the optimization process is stopped if no significant improvement occurs in the value of the function after 15 successive iterations. This explains the variation in the number of evaluations of the objective function for this technique.

The computational code of the IDEP was developed by the authors in C++ and simulations were solved by using a computer Intel® Core^{TM} i5-430M Processor and 6 GB of RAM. The parameters used in the IDEP were: population with 64 subjects divided into 4 processors, 50 generations, ED/best/1/bin strategy, difference factor F = 0.8 and crossover probability CR = 0.6.

The adopted criterion to stop the IDEP was the maximum number of generations of the population (which is 50 iterations). Thus, all cases studied were performed with 16000 evaluations of the objective function.

In the following tables the optimal values found by Oliveira (2006) applying Differential Evolution (DE) and the optimal values obtained using Improved Differential Evolution implemented in parallel (IDEP) are summarized.

Table 1: Optimal values considering the Weighting Objectives Method.

Weight Coef.	Algorithm	Volume [u.v.]	Dexterity	Stiffness [u.s.]	$[d_3\ d_4\ r_2\ r_3\ \alpha_2\ \alpha_3]$	Time	N.E.*
$w_1 = 0.8$	DE	3691.67	1.15	254.05	[3.00 3.00 3.00 3.00 3.00 -90.00 -75.43]	17.48 h	435
$w_2 = w_3 = 0.1$	IDEP	3678.19	1.18	255.30	[3.00 3.00 3.00 3.00 3.00 48.59 74.10]	14.15 min	16000
$w_2 = 0.8$	DE	3608.81	1.01	249.44	[3.00 3.00 3.00 3.00 3.00 89.48 -90.00]	17.46 h	540
$w_1 = w_3 = 0.1$	IDEP	3359.80	1.04	264.51	[3.00 3.00 3.00 3.00 3.00 90.00 32.93]	14.30 min	16000
$w_1 = w_2 = 0.1$	DE	2905.83	1.17	283.51	[3.00 3.00 3.00 3.00 3.00 45.36 -73.29]	16.51 h	465
$w_3 = 0.8$	IDEP	2751.82	1.39	290.69	[3.00 3.00 3.00 3.00 3.00 88.75 24.18]	14.36 min	16000
$w_1 = 0.5$	DE	3609.13	1.01	249.93	[3.00 3.00 3.00 3.00 3.00 90.00 -90.00]	10.09 h	300
$w_2 = w_3 = 0.25$	IDEP	3646.38	1.17	265.42	[3.00 3.00 3.00 3.00 3.00 -11.10 -74.22]	14.38 min	16000
$w_2 = 0.5$	DE	2846.61	1.02	214.98	[3.00 3.00 3.00 3.00 3.00 -81.55 -90.00]	7.77 h	240
$w_1 = w_3 = 0.25$	IDEP	3556.31	1.04	261.85	[3.00 3.00 3.00 3.00 3.00 79.83 89.10]	14.38 min	16000
$w_1 = w_2 = 0.25$	DE	3432.88	1.01	261.99	[3.00 3.00 3.00 3.00 3.00 68.78 90.00]	16.01 h	450
$w_3 = 0.5$	IDEP	3499.26	1.14	271.17	[3.00 3.00 3.00 3.00 3.00 89.18 20.01]	14.13 min	16000
$w_1 = w_2 =$	DE	3601.96	1.01	249.97	[3.00 3.00 3.00 3.00 3.00 89.55 90.00]	11.78 h	345
$w_3 = 1/3$	IDEP	3617.24	1.15	267.07	[3.00 3.00 3.00 3.00 3.00 70.8989 -7.46]	14.01 min	16000

* Number of evaluations of the objective function.

Table 2: Optimal values considering the Global Criterion Method.

Metric	Algorithm	Volume [u.v.]	Dexterity	Stiffness [u.s.]	$[d_3\ d_4\ r_2\ r_3\ \alpha_2\ \alpha_3]$	Time	N.E.*
L_1	DE	3689.50	1.20	262.60	[3.00 3.00 3.00 3.00 3.00 90.00 72.15]	29.09 h	750
	IDEP	3689.50	1.19	260.91	[3.00 3.00 3.00 3.00 3.00 14.15 57.86]	14.01 min	16000
L_2	DE	3681.31	1.21	263.17	[3.00 3.00 3.00 3.00 3.00 90.00 69.97]	25.84 h	750
	IDEP	3654.82	1.27	265.34	[3.00 3.00 3.00 3.00 3.00 45.67 -30.20]	14.01 min	16000
L_{2R}	DE	3436.38	1.01	262.68	[3.00 3.00 3.00 3.00 3.00 68.58 89.22]	14.82 h	450
	IDEP	3609.40	1.00	251.60	[3.00 3.00 3.00 3.00 3.00 39.18 89.52]	14.08 min	16000
L_3	DE	3679.75	1.15	257.88	[3.00 3.00 3.00 3.00 3.00 -89.26 67.62]	26.27 h	750
	IDEP	3686.07	1.14	255.89	[3.00 3.00 3.00 3.00 3.00 73.78 -55.29]	13.97 min	16000
L_{3R}	DE	3466.22	1.04	262.92	[3.00 3.00 3.00 3.00 3.00 69.55 87.59]	16.54 h	495
	IDEP	3266.69	1.01	259.71	[3.00 3.00 2.96 3.00 2.89 90.00 89.48]	14.31 min	16000

* Number of evaluations of the objective function.

It is worth noting that the optimal results present in Tab. (2) are strongly dependent on the weighting coefficients.

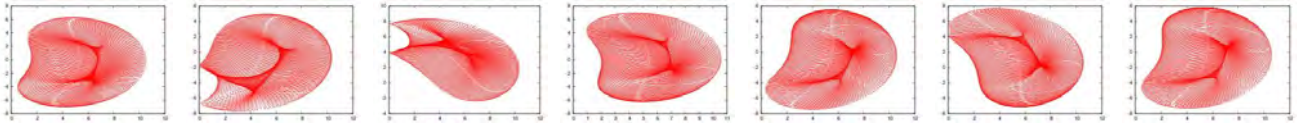
The ideal values (optimal value for each objective function considered separately) calculated for the workspace volume, dexterity and system stiffness are: V_{ideal}=3689.507 [u.v.], D_{ideal}=1.0035 and S_{ideal}=293.2703 [u.s.], respectively.

Analyzing the results shown in the tables, one can conclude that the methods are effective to solve the problem since the results approach the ideal values.

Comparing the results obtained with DE and IDEP, it is possible to observe that are very similar. However, for the studied problem, the IDEP method reached the optimum faster. This analysis should be very careful, these results do not mean that this method is always the best.

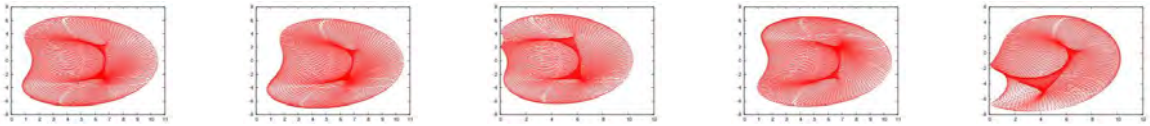
ISSN 2176-5480

Figures 5 and 6 show the graph of the cross-sectional area of the workspace of the robot manipulator considering the Weighting Objectives Method and the Global Criterion Method, respectively.



(a) $w_1 = 0.8$, $w_2 = w_3 = 0.1$ (b) $w_2 = 0.8$, $w_1 = w_3 = 0.1$ (c) $w_3 = 0.8$, $w_1 = w_2 = 0.1$ (d) $w_1 = 0.5$, $w_2 = w_3 = 0.25$ (e) $w_2 = 0.5$, $w_1 = w_3 = 0.25$ (f) $w_3 = 0.5$, $w_1 = w_2 = 0.25$ (g) $w_1 = w_2 = w_3 = 1/3$

Figure 5: Cross-sectional area of the workspace of the robot manipulator considering the Weighting Objectives Method



(a) Metric L_1 (b) Metric L_2 (c) Metric L_{2R} (d) Metric L_3 (e) Metric L_{3R}

Figure 6: Cross-sectional area of the workspace of the robot manipulator considering the Global Criterion Method

6. CONCLUSIONS

In this contribution, two evolutionary techniques, the Differential Evolution algorithm and the Improved Differential Evolution algorithm (implemented in parallel) were used to solve a multi-objective problem. An optimal manipulator design was presented to illustrate the methodology studied here.

Successful numerical applications have demonstrated the efficiency of these techniques. The preliminary results seem to indicate that the IDEP is faster in achieving the optimal solution and that it's highly parallelizable. Furthermore, the need for communication between the processors is minimum, thus ensuring a great decrease in execution time of the program in relation to the sequential algorithm.

For future work we intend to use the Improved Differential Evolution optimization method implemented in parallel to find a solution of large linear systems, using the model presented by Purcina (2010).

7. ACKNOWLEDGEMENTS

The authors acknowledge the financial support provided by FAPEMIG (Fundação de Amparo à Pesquisa de Minas Gerais) and CAPES.

8. REFERENCES

Abdel-Malek, K., Yeh, H.J. and Othman, S., 2000. "Understanding voids in the workspace of serial robot manipulators". In *Proceedings Pf 23rd ASME, Design Engineering Technical Conference*. Baltimore, Maryland.

Bergamaschi, P.R., Nogueira, A.C. and Saramago, S.F.P., 2006. "Design and optimization of 3r manipulators using the workspace features". *Applied Mathematics and Computation*, Vol. 172, pp. 439–463.

Bergamaschi, P.R., Saramago, S.F.P. and Coelho, L.S., 2008. "Comparative study of sqp and metaheuristics for robotic manipulator design". *Applied Mathematics and Computation*, Vol. 58, pp. 1396–1412.

Brandão, M.A.L., Doricio, J.L., Lobato, F.S. and Saramago, S.F.P., 2012. "A comparative study using shuffled complex evolution and the differential evolution applied to robotic manipulator design". In *10^{th} World Congress on Computational Mechanics, Computational Mechanics 2012 Proceedings*. São Paulo, Brazil.

Brandão, M.A.L., Oliveira, L.S. and Saramago, S.F.P., 2011. "Técnicas de evolução diferencial com conjuntos embaralhados". In *21 POSMEC - Simpósio do Programa de Pós-Graduação em Engenharia Mecânica da Universidade Federal de Uberlândia*. Uberlândia, Brazil.

Bujok, P., 2011. "Parallel models of adaptive differential evolution based on migration process". *Aplimat - Journal of Applied Mathematics*, Vol. 4, No. 2.

Ceccarelli, M., 1996. "A formulation for the workspace boundary of general n-revolute manipulators". *IFToMM Journal of Mechanism and Machine Theory*, Vol. 31, No. 5, pp. 637–646.

Ceccarelli, M. and Lanni, C., 2004. "A multi-objective optimum design of general 3r manipulators for prescribed workspace limits". *Mechanism and Machine Theory*, Vol. 39, pp. 119–132.

Chong, C.K., S, M.M., S, D., S, S.M., W, C.Y. and E, C.L., 2012. "Improved differential evolution algorithm for parameter

ISSN 2176-5480

estimation to improve the production of biochemical pathway". *International Journal of Interactive Multimedia and Artificial Intelligence. Special Issue on Distributed Computing and Artificial Intelligence*, Vol. 1, pp. 22–29.

Coelho, L.S., Souza, R.C.T. and Mariani, V.C., 2009. "Improved differential evolution approach based on cultural algorithm and diversity measure applied to solve economic load dispatch problems". *Mathematics and Computers in Simulation*, Vol. 79, pp. 3136–3147.

Das, S., Konar, A. and Chakraborty, U.K., 2005. "Two improved differential evolution schemes for faster global search". In *GECCO' 05*. Washington, DC, USA.

Duan, Q.A., Gupta, V.K. and Sorooshian, S., 1993. "A parallel differential evolution algorithm". *Journal of Optimization Theory and Applications*, Vol. 76, No. 3.

Kwedlo, W. and Bandurski, K., 2006. "A parallel differential evolution algorithm". In *Parallel Computing in Electrical Engineering. PAR ELEC 2006*.

Lanni, C., Saramago, S.F.P. and Ceccarelli, M., 2002. "Optimal design of 3r manipulators using classical techniques and simulated annealing". *Revista Brasileira de Ciências Mecânicas*, Vol. 24, No. 4, pp. 293–301.

Li, R., Xu, L., Shi, X.W., Zhanh, N. and Lv, Z.Q., 2011. "Improved differential evolution strategy for antenna array pattern synthesis problems". *Progress In Electromagnetics Research*, Vol. 113, pp. 429–441.

Mehablia, A., 2010. "An improved differential evolution algorithm and its application in reaction kinetic parameters estimation". In *European Conference of Chemical Engineering*.

Oliveira, G.T.S., 2006. *Estudo e Aplicações da Evolução Diferencial*. Master's thesis, Universidade Federal de Uberlândia, Uberlândia, MG, Brasil.

Oliveira, L.S. and Saramago, S.F.P., 2010. "Multiobjective optimization techniques applied to engineering problems". *Journal of the Brazilian Society of Mechanical Sciences and Engineering (Impresso)*, Vol. XXXII, pp. 94–104.

Purcina, L.A., 2010. *Técnicas de Otimização Evolutivas aplicadas à Solução de Grandes Sistemas Lineares*. Ph.D. thesis, Universidade Federal de Uberlândia, Uberlândia, MG, Brasil.

Raupp, F.M.P., Fampa, M.C.H. and Melo, W.A.X., 2010. "Evolução diferencial aperfeiçoada para otimização contínua restrita". In *XLII SBPO*. Bento Gonçalves-RS.

Saramago, S.F.P., Ottaviano, E. and Ceccarelli, M., 2002. "A characterization of the workspace boundary of three-revolute manipulators". In *Design Engineering Technical Conferences, Proceedings of DETC-02, ASME 2002*. Montreal, Vol. 1, pp. 34342–34352.

Tasoulis, D.K., Pavlidis, N.G., Plagianakos, V.P. and Vrahatis, M.N., 2004. "Parallel differential evolution". In *Congress on Evolutionary Computation (CEC 2004)*. pp. 2023–2029.

Vargas, J.R.G., Reynoso, L.V.J.M. and Mier-Maza, R., 1992. "Diseño de un manipulador industrial para aplicaciones de limpieza en subestaciones eléctricas". In *Centro Metropolitano de Investigación en Mecatrónica*. ITESM Querétaro.

9. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.