Comparison of a concurrent and a sequential optimization methodologies for serial manipulators using metaheuristics

Salvador Botello-Aceves*, S. Ivvan Valdez‡, Héctor M. Becerra* and Eusebio Hernandez†
* Center for Research in Mathematics (CIMAT), Guanajuato, Mexico.
* University of Guanajuato, Division of Engineering, Palo Alto, Salamanca, Mexico.
† Instituto Politecnico Nacional, SEPI ESIME Ticoman, Ciudad de Mexico, Mexico.
Email: {salvador.botello}@cimat.mx, si.valdez euhernandezm@ipn.mx

***Abstract*—A concurrent optimum design of a manipulator is to find the best geometrical and control parameters in the same optimization process. One of the main contributions of this article is a concurrent method for optimum design and its comparison versus a sequential one, where the last requires several optimization stages. Besides, we statistically compare three metaheuristics: the Omnioptimizer, the CMA-ES, and the BUMDA for the two methodologies, in order to elucidate directions about which metaheuristic performs the best for this kind of problem, and whether it must be used in a sequential or concurrent fashion. These metaheuristics are compared with reported results in the specialized literature. In addition, we perform an analysis of results to statistically determine relations between the parameters and the objective function, as a consequence, we found the most impacting parameters to the manipulator performance.**

***Index Terms*—Genetic algorithms, Evolutionary computation, Optimization methods, Design optimization, Concurrent design, Sequential design.**

I. Introduction

The *concurrent* optimal design of a manipulator is defined as *finding optimal structural and control parameters for a given objective function during the same optimization process, which is dependent on the kinematic or dynamic model of the mechanism*. The design parameters can be links lengths and control gains among others. The concurrent optimization problem could also be addressed as two disjoint problems, firstly searching for the optimal structure parameters, and secondly, for optimal control parameters. This way of addressing the problem is referred here as *the sequential*. The contribution of this paper is two-fold: First, we introduce and compare the sequential and concurrent methodologies. To the best of our knowledge, there is not other paper approaching the **concurrent** optimal design of a six degrees of freedom (DOF) manipulator in an automatic fashion such as this proposal does. Second, we compare three metaheuristics from three different families of algorithms, and we show which one is better suited for the optimization of serial manipulators.

Metaheuristics are powerful tools to solve optimization problems and their application in engineering optimization is well established. For instance, in [1], a survey on applications to structural optimization is presented and in [2], common applications in robotics are discussed, contributing with a taxonomy of robotic optimization problems. According to it, this article addresses a problem in the class of **individual robot layer**, in which the authors also identify the motion and path planning problems. Nevertheless it is an extended review, the structure design and the concurrent optimization problems are not considered. In this vein, our proposal contributes with discussion and directions about these problems and the appropriate metaheuristics to approach them. We differentiate three types of optimization problems, which can be classified as *static, kinematic and dynamic*, according to the mathematical model required to numerically simulate the systems. *In the static optimization problems* the governing model is a set of algebraic kinematic equations expressed as homogeneous transformations. Very often, static problems are formulated as maximizing the workspace [3], [4], [5] or reaching a set of point landmarks, considering joint constraints, singularities avoidance [6], stiffness [7] and dexterity [8]. For instance, workspace maximization is addressed in [9] using the Teaching-Learning-Based Optimization algorithm (TLBO), and it is compared with several evolutionary algorithms and Sequential Quadratic Programming (SQP), being SQP the worst performed. Other comparisons of local and global optimizers applied to robotics have been reported similar competitive results of global optimizers [3], [10], [11].

Kinematic optimization problems require to solve a first-order differential equation that represents relationships among velocities. Typically, the manipulator's end-effector must track a trajectory imposed by the user, and the optimization problem consists in minimizing the error between the trajectory and the position and orientation of the end-effector. It involves solving the differential kinematics of the manipulator, applying collision avoidance, position and velocity constraints [12]. It is common to use the classical Proportional-Integral-Derivative (PID) control [13] or its P or PD variants [14]. Some authors focused in optimizing, only, the structural parameters [15], and others on the control parameters [16], [13]. In [17], a memetic algorithm for optimum design of the control parameters of a

*The first and third authors were supported in part by Conacyt [grant 220796].

serial manipulator is proposed. Kinematics is considered by defining a set of target points in the workspace which must be reached by the end-effector. Similarly, in [13], a simultaneous design optimization of a planar manipulator and a non-linear gain for a PD controller is presented. They combine concepts from Genetic Algorithms (GA) and Evolution Strategies (ES).

In *dynamic optimization problems* a second order differential equation must be solved, since the model to simulate the manipulator represents relationships at the level of accelerations. A common dynamic robotic task is minimizing the error between the executed and the desired joints trajectories or to minimize the energy consumed by every joint through the whole robot's motion, subject to collisions and dexterity constrains. In [18], a method for energy-consumption optimization in function of energetic losses of a PUMA 560 serial manipulator and a parallel kinematic machine is presented. For the case of a two-link planar manipulator, an ES is used for simultaneous optimization of design parameters and control gains of a non-linear PD controller in [19]. In a similar vein, the optimization of a PID controller of a two-link planar manipulator through the Non-dominated Sorting Genetic Algorithm II (NSGA-II) is presented in [20]. In [21], a methodology to target kinematic and dynamic performance in a sequential manner is proposed, nevertheless it is not compared with other metaheuristics or with a concurrent methodology, issues that are in the scope of our contributions.

This article presents three optimization models, one for each category: static, kinematic and dynamic. The three are only used by the sequential methodology, where the output of a first model is the input for a second, and so on. We focus on comparing the dynamic optimization model under both the sequential and concurrent methodologies. This is the problem of interest because it summarizes the performance of the structure (links lengths) and control parameters, which are the optimization variables in both methodologies. Notice that the concurrent optimization problem is more complex than the sequential, each set of lengths has an optimal set of control gains. Hence, the search space for the concurrent problem (lengths **and** control gains) is higher than in the sequential stages (lengths **or** control parameters). The concurrent methodology aims to find the best overall design, while the sequential aims to find a set of lengths and then the optimal control gains for those lengths, thus if the lengths are not optimal, neither the control parameters. On the one hand, the subproblems approached by the sequential methodology are simpler than the concurrent, on the other hand the sequential methodology is more prone to be trapped in a local minimum.

The organization of the paper is as follows: Section II describes the proposed evolutionary design methodologies: sequential and concurrent. In Section III, we introduce the three optimization models used for each design methodology. Results for two cases of study, a comparative analysis and discussion are presented in Section IV. Finally, concluding remarks are given in Section V.

II. Evolutionary design methodology

An objective function that defines the performance of a manipulator with respect to its lengths and control parameters is, by nature, highly non-linear and multimodal, considering that it must encompass solving a set of nonlinear differential equations, and that changing a link length significantly changes the manipulator's kinematics and dynamics. Thus, an adequate optimizer must be chosen accordingly. For instance, we require a continuous optimizer for non-linear and non-convex optimization. Commercial software, i.e., ®Matlab, include a set of widely tested and well performed optimization software (https://www.mathworks.com/products/optimization.html). On the other hand, the methods we consider here, currently are in the testing stage by the research community, they are considered cutting edge methods in the state of the art, which possibly, will be implemented in commercial software in a near future. It is worth mentioning that neither commercial solvers nor methods under development guarantee to reach global optima in the kind of addressed problem, however, the last can obtain adequate solutions for the problem. This section presents three metaheuristics for non-linear, multimodal problems, that are compared in the manipulators' design, and introduces two generic design methodologies: the sequential and the concurrent.

A. Evolutionary algorithms

Evolutionary Algorithms (EAs) are based on the principles of biological evolution. They evolve a population of individuals (candidate solutions), by applying reproduction and variation operators on a selected set which is biased to the best solutions, the intention is to produce fitter offspring for the next generation. The offspring replace the current population, usually, preserving the best solution through generations, this is called elitism. The process is repeated until a stopping criterion is met. The EAs used in this paper can be classified into three families: Evolution Strategies [22], Genetic Algorithms [23] and Estimation of Distribution Algorithms (EDAs) [24], all of them classified as evolutionary algorithms, but each of them with a particular way of working. The *Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES)* [22], uses a multivariate normal distribution to search for the optimum. A set of parent solutions is used to determine the size, position and orientation of a normal distribution used to sample the children candidate solutions. The orientation of the multivariate normal model aligns the search to promising regions. The *Omnioptimizer* [23] is the implementation of a GA, it selects the most promising individuals by using the binary tournament operator, then, individuals in the selected set are re-combined using the Simulated Binary Crossover (SBX). Exploration on the search space is maintained via a polynomial mutation operator. The *Boltzmann Univariate Marginal Distribution Algorithm (BUMDA)* [24] estimates a probability distribution by using the best candidates, the better a candidate is, the higher the weight of such candidate in the estimation formula. Thus, the resulting probability functions favor to sample the best regions already known. It requires a single parameter (population size), its complexity per iteration

is $O(MN\log(N))$, for M dimensions and N population size, while CMA-ES is $O(M^2N\log(N))$. Nevertheless, Omnioptimizer is, $O(MN\log(N))$ for mono-objective problems, the total number of operations is greater than that of the BUMDA.

B. Design optimization methodologies

The optimization problems in this paper are addressed as **mono-objective**. We introduce two optimization methodologies. A flowchart of the ***sequential methodology*** is shown in Fig. 1, it works as follows:

1) Select an optimization algorithm from the following: a) BUMDA, b) CMA-ES, c) Omnioptimizer. **2)** The first subproblem is to optimize the structure lengths for maximizing the workspace volume. The output is the link lengths and they are fixed for the next step. **3)** For the second optimization subproblem, given a desired trajectory for the end-effector, optimize kinematic control parameters for minimizing the tracking error. The approximated optimum control parameters are used to compute feasible joint position and velocity profiles. Thus, it maps the trajectory of the end-effector to positions and velocities of the joints, obtained as a table of discrete values; the kinematic control parameters are dismissed. **4)** Optimize control parameters of joints for minimizing energy consumption and joint errors with respect to the position and velocity profiles from the previous optimization step.

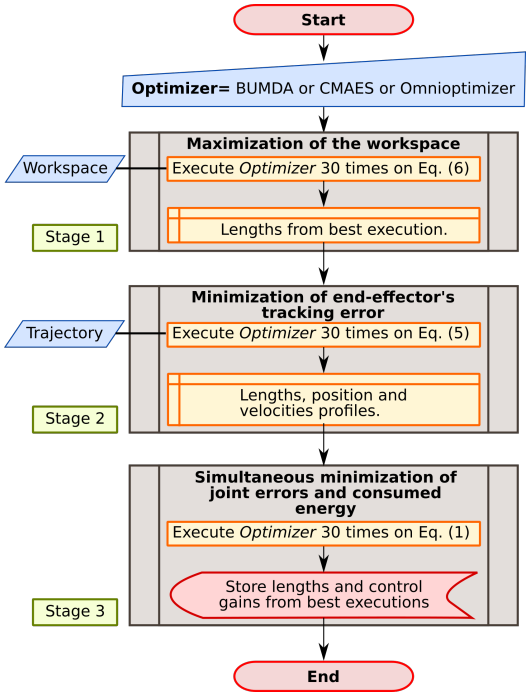

Fig. 1. Flowchart of the sequential methodology.

On the other hand, the flowchart of the ***concurrent methodology*** is shown in Fig. 2, which works as follows:

1) Select an optimization algorithm from the following: a) BUMDA, b) CMA-ES, c) Omnioptimizer. **2)** Given a desired trajectory of the joint variables (input), optimize lengths

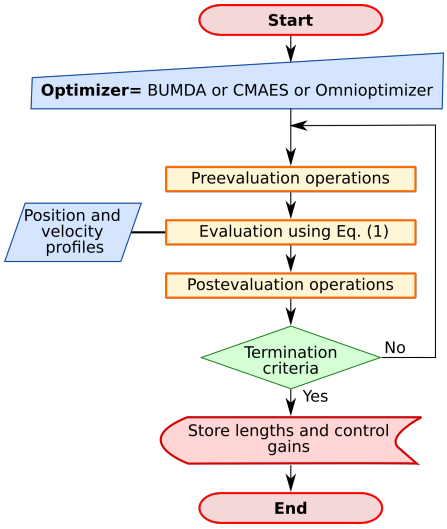

Fig. 2. Flowchart of the concurrent methodology.

of the structure and control parameters (outputs), for minimizing joint errors and energy consumption via a weighted function. Thus, preevaluation operations in Fig. 2, are, for instance, crossover and mutation in the Omnioptimizer, and sampling from a probability distribution in the CMA-ES and BUMDA. The postevaluation operations are selection of the parents to yield the next generations in the Omnioptimizer, and for updating the probability distribution in the CMA-ES and BUMDA, and elitism in all the algorithms.

In both optimization methodologies the outputs are approximations to the optimum structure lengths and control parameters of joint controllers. Both methods are executed 30 times and their best outputs are compared. The concurrent methodology can be directly applied if joint desired trajectories are known, otherwise the process can be also aided by an initial step to map the desired trajectory of the end-effector to position and velocity profiles of the joints. Thus, this initial step reduces the effort of computing the inverse kinematics of the manipulator for each point in the trajectory of the end-effector and ensures that the desired positions and velocities can actually be reached by the mechanism.

III. Optimization Models for Optimum Design

To precisely define the variables involved in the optimization models, consider a manipulator with n joints, l links, m actuators and d degrees of freedom of the end-effector. We define $\boldsymbol{\theta} \in \mathbb{R}^{\boldsymbol{m}}$ as actuated joint variables and $\boldsymbol{\phi} \in \mathbb{R}^{\boldsymbol{n-m}}$ as passive joint variables. The mechanism is defined by a set of design parameters $\boldsymbol{\alpha} = [\boldsymbol{l_1}, ..., \boldsymbol{l_l}, \boldsymbol{k_p^{(1)}}, ..., \boldsymbol{k_p^{(m)}}, \boldsymbol{k_d^{(1)}}, ..., \boldsymbol{k_d^{(m)}}, \boldsymbol{k_i^{(1)}}, ..., \boldsymbol{k_i^{(m)}}] \in \mathbb{R}^{\boldsymbol{p}}$, in this case, the lengths of the links and the control parameters.

We introduce three objective functions that are used in the concurrent and sequential methodologies: *1) Simultaneous minimization of joint errors and consumed energy* is the main problem addressed in the paper, and corresponds to a dynamic

optimization problem. *2) Minimization of end-effector's tracking error* is a kinematic optimization problem that is used in the concurrent and sequential methodologies. The optimization variables are the kinematic control parameters, nevertheless, they are finally dismissed. We use this optimization process as a high level control layer that generates position and velocity profiles for each joint variable inputted as desired trajectories to the previous dynamic optimization problem. *3) Maximization of the workspace* is a static optimization problem *only* used as the first stage in the *sequential methodology*. The output of this optimization process are the lengths delivered by the sequential methodology. The reader can consult [25] for a deeper explanation, discussion and additional study cases which use these models.

A. Simultaneous minimization of joint errors and energy

This is the main optimization model introduced in this paper, it is a weighted sum of time integrals. The first term depends on the absolute error between the desired and the current position of each joint. The second is the integral of the absolute torque applied to each joint. The last term is a penalization to the torque gradients, whose aim is to reduce sudden changes in the trajectory.

$$\begin{aligned}\min_{\boldsymbol{\alpha}} \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}) = \boldsymbol{\omega_1} \sum_{i=1}^{m} \int_{t_0}^{t_n} |e^{(i)}(t)| \partial t + \\ \boldsymbol{\omega_2} \sum_{i=1}^{m} \int_{t_0}^{t_n} |\boldsymbol{\tau}^{(i)}(t)| \partial t + \boldsymbol{\lambda} \sum_{i=1}^{m} \int_{t_0}^{t_n} |\boldsymbol{\Delta\tau}^{(i)}(t)| \partial t,\end{aligned} \tag{1}$$

$$\boldsymbol{S.\,T.}\quad \boldsymbol{\theta}_i^{\min} \leq \boldsymbol{\theta}_i \leq \boldsymbol{\theta}_i^{\max} \text{ and } \boldsymbol{\phi}_j^{\min} \leq \boldsymbol{\phi}_j \leq \boldsymbol{\phi}_j^{\max}, \tag{2}$$

where $\boldsymbol{i = 1, ..., m}$, $\boldsymbol{j = 1, ..., n - m}$, $\boldsymbol{\omega_1}$, $\boldsymbol{\omega_2}$ and $\boldsymbol{\lambda}$ are weights of each term, the used values are described in each experiment. The expressions in Eq. 2 represent joint limits. We assume that they are set in such a way that self-collisions are avoided. Position and velocity joint errors are given by:

$$\mathbf{e}(t) = \boldsymbol{\theta}(t) - \boldsymbol{\theta}^D(t),\ \dot{\mathbf{e}}(t) = \dot{\boldsymbol{\theta}}(t) - \dot{\boldsymbol{\theta}}^D(t), \tag{3}$$

with $\boldsymbol{\theta}^D(t) \in \mathbb{R}^m$ and $\dot{\boldsymbol{\theta}}^D(t) \in \mathbb{R}^m$ being desired position and velocity profiles. Regarding the second term of Eq. 1, we assume that each actuated joint is commanded by a classical PID controller, such that the vector of torques is given by:

$$\boldsymbol{\tau}(t) = \mathbf{k}_p \mathbf{e}(t) + \mathbf{k}_d \dot{\mathbf{e}}(t) + \mathbf{k}_i \int_{t_0}^{t_n} \mathbf{e}(t) \partial t \in \mathbb{R}^m, \tag{4}$$

where $\mathbf{k}_p$, $\mathbf{k}_d$ and $\mathbf{k}_i$ are diagonal matrices with the proportional, derivative and integral control parameters. Regarding the third term of the objective function, $\boldsymbol{\Delta\tau}^{(i)}(t)$ is the numerical torque gradient for each i actuator, defined as $\boldsymbol{\Delta\tau}^{(i)}(t) = \boldsymbol{\tau}^{(i)}(t) - \boldsymbol{\tau}^{(i)}(t-1)$. This model is used in the concurrent methodology and in the last stage of the sequential methodology. In the *concurrent methodology* the optimization variables are the set of link lengths and dynamic control parameters $\boldsymbol{\alpha} = [\boldsymbol{l_1, ..., l_l, k_p^{(1)}, ..., k_p^{(m)}, k_d^{(1)}, ..., k_d^{(m)}, k_i^{(1)}, ..., k_i^{(m)}}]$, while for the last optimization stage of the *sequential methodology* the optimization variables are the dynamic control parameters maintaining the lengths constant.

B. Minimization of end-effector's tracking error

This model is for minimizing the error between a desired and the current trajectory followed by the robot's end-effector, by optimizing kinematic control parameters. However, we use it as a mean to generate position and velocity profiles delivered by the approximated-optimal control parameters. They are used in the concurrent and sequential methodologies. We consider that each DOF of the manipulator's end-effector is under a proportional-derivative (PD) control action. Hence, the objective function is the integral of the absolute value of the control signal, for shorting $\boldsymbol{e^{(i)}(t) = e^i}$ and $\boldsymbol{\dot{e}^{(i)}(t) = \dot{e}^i}$, as follows:

$$\min_{\boldsymbol{\alpha}} \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}) = \sum_{i=1}^{d} \left(\boldsymbol{\kappa_p^{(i)}} \int_{t_0}^{t_n} |\boldsymbol{e^i}| \partial t + \boldsymbol{\kappa_d^{(i)}} \int_{t_0}^{t_n} |\boldsymbol{\dot{e}^i}| \partial t, \right), \tag{5}$$

subject to (2). Where $\boldsymbol{\kappa_p^{(i)}}$ and $\boldsymbol{\kappa_d^{(i)}}$ are proportional and derivative control gains, respectively, which are the optimization variables, $\boldsymbol{\alpha} = [\boldsymbol{\kappa_p^{(1)}, ..., \kappa_p^d, \kappa_d^{(1)}, ..., \kappa_d^{(d)}}]$. The functions $\mathbf{e}(t) = \mathbf{X}(t) - \mathbf{X}^D(t) \in \mathbb{R}^d$ and $\dot{\mathbf{e}}(t) = \dot{\mathbf{X}}(t) - \dot{\mathbf{X}}^D(t) \in \mathbb{R}^d$ are translation and velocity errors, respectively, $\mathbf{X}(t)$ and $\dot{\mathbf{X}}(t)$ are current translation and velocity coordinates (including orientation in both cases) of the manipulator's end-effector. Likewise, $\mathbf{X}^D(t)$ and $\dot{\mathbf{X}}^D(t)$ are desired translation and velocity coordinates given by the desired trajectory.

C. Maximization of the workspace

This objective function is only used by the sequential methodology with the purpose of finding optimum link lengths that are fixed for subsequent optimization stages. The problem is written as a minimization problem, in order to be consistent in all the objectives and algorithms. A workspace $\boldsymbol{W}$ is discretized by a set of $\boldsymbol{r}$ points in $\boldsymbol{d}$ dimensions, $\mathbf{X} \in \mathbb{R}^{r \times d}$.

$$\min_{\boldsymbol{\alpha}} \boldsymbol{\mathcal{F}}(\boldsymbol{\alpha}) = 1 - \boldsymbol{\mathcal{W}}(\boldsymbol{\alpha}), \tag{6}$$

Subject to (2), where the link lengths are the optimization variables $\boldsymbol{\alpha} = [\boldsymbol{l_1, ..., l_l}]$. We assume that joint limits are set such that self-collisions are not possible. In Eq. 6 $\boldsymbol{\mathcal{W}(\alpha)}$ is the percentage of reached points in $\boldsymbol{W}$.

D. Considerations about implementing the objective functions

As can be seen in Eqs. (1) and (5), absolute functions are used to correctly compute the accumulated values of errors and torques along the time. Indeed, similar results can be obtained by using quadratic functions. The numerical simulation of robot with explicit iterative methods such as the classical 4th order Runge-Kutta used in this article, returns the torque, velocities, positions, and corresponding errors discretized with the same time step. Finally, if the integrals in the objective functions are computed with methods of higher order, then the

values must be very similar among methods and it is expected that the integrals must converge to the real value.

E. Multi-objective vs mono-objective approaches

As aforementioned, the previous optimizations problems are addressed as mono-objective. Regarding the case of simultaneous minimization of joint errors and energy, the problem can be addressed in two fashions: as a mono-objective approach *a priori* assuming that adequate weights $\boldsymbol{\omega_1}$, $\boldsymbol{\omega_2}$ and $\boldsymbol{\lambda}$ have been chosen, or as a multi-objective problem with three objectives considering that *a posteriori*, a Decision Maker (DM) will chose an adequate solution from an optimal set (Pareto set). Considering that two of the optimizers tested in this article are not equipped for multi-objective optimization (BUMDA and CMA-ES), and that the multi-objective approach comprises additional issues such as convergence to the real Pareto front, spreading and coverage of the solution space, and DM support for the final decision, we have chosen addressing a mono-objective problem, and to use these results as a reference for a future multi-objective investigation which considers all the mentioned issues and other studied in specialized literature **??**.

IV. Results of case studies

We present two case studies, a 2-DOF 2-link serial manipulator and a 6-DOF anthropomorphic serial manipulator. For the **2-DOF 2-link serial manipulator** the purposes of the experiments are the following: *a)* Show that the selected optimizers are competitive and even superior than other metaheuristic reported in specialized literature, by comparing the results of the metaheuristics used in this paper with reported results for a 2-DOF manipulator. *b)* Contrast the sequential and concurrent methodologies in a simple case, executing both methodologies with the best performed metaheuristic. The 2-DOF manipulator has 2 actuated joints ($\boldsymbol{n = m = 2}$) to control the 2 DOF ($\boldsymbol{d = 2}$). The structure parameters are 2 link lengths $(\boldsymbol{l_1, l_2})$ and the control parameters are 6 $(\boldsymbol{k_p^{(1)}, k_p^{(2)}, k_d^{(1)}, k_d^{(2)}, k_i^{(1)}, k_i^{(2)}})$.

For the **6-DOF anthropomorphic manipulator** the purposes of the experiments are the following: *a)* Compare the sequential versus the concurrent methodology, using the best performed metaheuristic, in a more complex case study. *b)* Show how to take advantage of the posterior information (after several executions of the methodologies), analyzing the correlations among variables and objective function to elucidate which variables have the most impact on the manipulator's performance. The anthropomophic manipulator has 6 actuated joints ($\boldsymbol{n = m = 6}$) to control its 6 DOF ($\boldsymbol{d = 6}$). The structure parameters are 4 link lengths $(\boldsymbol{l_1, ..., l_4})$ and the dynamic control parameters are 18 $(\boldsymbol{k_p^{(1)}, ..., k_p^{(6)}, k_d^{(1)}, ..., k_d^{(6)}, k_i^{(1)}, ..., k_i^{(6)}})$.

We use two kinds of stopping criteria for the optimizers evaluation. The first is the number of evaluations, which yields to fair comparisons with similar computational cost for all algorithms. All algorithms stop at a maximum of $\boldsymbol{1e4 \cdot p}$ evaluations, for $\boldsymbol{p}$ optimization variables. The second is based on the exploration capability of the algorithms, for instance, the CMA-ES stops if the difference between two consecutive generations is less than $\boldsymbol{1.0e}$-20, tiny changes not only are neglected improvements of the performance, but, they indicate that the probability of exploring a different region than the current is almost null. The same applies for the BUMDA, it requires to compute all marginal variances, thus, if the maximum marginal variance is less than $\boldsymbol{1.0e}$-12, then, it is stopped. The Omnioptimizer stops when it reaches $\boldsymbol{40}$ generations, as suggested in [26]. This is a suggested and tested criterion in literature, since this optimizer does not have a variance measure as the two others. Supplementary material is provided in the following address: https://drive.google.com/file/d/1iLVCJet3DQVZbE3C75CF3BX9YArKKfS3/

A. Case study: Two-link serial manipulator

A representation of the mechanism is shown in Fig. 3. This simple robot has been considered as a benchmark in the literature [19], [20], [26].

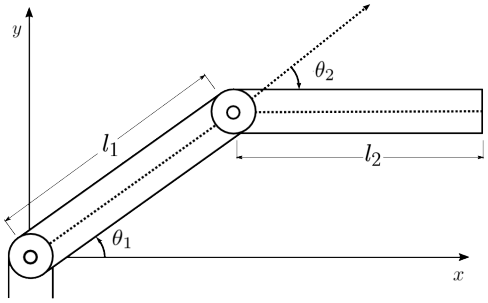

Fig. 3. Case study: Two-link serial manipulator. Scheme of its structure.

We address the optimization problem presented in the objective function of Eq. 1, to initially optimize only the parameters of a PID control for each joint. The desired trajectory is given by $\boldsymbol{\theta_i^D(t) = \theta_i^D(t_0) + (6r_{time}^5 - 15r_{time}^4 + 10r_{time}^3)(\theta_i^D(t_f) - \theta_i^D(t_0))}$, where $\boldsymbol{r_{time} = t_0/t_f}$, $\boldsymbol{\theta_1^D(t_0) = \theta_2^D(t_0) = 0}$ for $\boldsymbol{t_0 = 0}$ when and $\boldsymbol{\theta_1^D(t_f) = \pi/2}$ and $\boldsymbol{\theta_2^D(t_f) = \pi/6}$ for $\boldsymbol{t_f = 2}$. The search range of the control parameters is [0, 400] for gains $\boldsymbol{k_p^{(i)}}$, $\boldsymbol{k_d^{(i)}}$ and $\boldsymbol{k_i^{(i)}}$, for $\boldsymbol{i = 1, 2}$. For links masses, lengths and inertial moments the ranges are [0.1,0.1], [0.8,0.4] and [0.64,0.016], respectively. In Eq. 1, we set $\boldsymbol{\omega_1 = \omega_2 = 0.5}$ and $\boldsymbol{\lambda = 0}$ which is denoted as $\boldsymbol{\mathcal{F}^*}$. Additionally, we will have the values of each term: the minimum energy ($\boldsymbol{\mathcal{F}_2}$), which is equivalent to set $\boldsymbol{\omega_1 = 0}$ and $\boldsymbol{\omega_2 = 1}$, and the minimum tracking error ($\boldsymbol{\mathcal{F}_1}$), which is equivalent to set $\boldsymbol{\omega_1 = 1}$ and $\boldsymbol{\omega_2 = 0}$. Hence, $\boldsymbol{\mathcal{F}^*}$ is the average of $\boldsymbol{\mathcal{F}_1}$ and $\boldsymbol{\mathcal{F}_2}$.

In the following subsection, we present the optimization results only for control parameters for the sake of comparing our method with the reference [26]. In the second subsection, we compare the sequential and concurrent methodologies for the optimization of structure lengths and control parameters.

1) Optimal design of a dynamic control: This experiment validates the use of the selected EAs with respect to other similar algorithms in the specialized literature. Table I, compares the results from [26] with the three algorithms used

in this paper. It shows that CMA-ES and Omnioptimizer return similar control parameters and values of $\mathcal{F}^*$. Although, BUMDA is not better than the other algorithms tested in this paper, it is better than the known reported results.

TABLE I
BEST SOLUTION FOR EQ. 1. OBJECTIVE=$\mathcal{F}^*$ FOR $\omega_1 = \omega_2 = 0.5$ AND $\lambda = 0$. $\mathcal{F}_1$ FOR $\omega_1 = 1$ AND $\omega_2 = 0$, AND $\mathcal{F}_2$ FOR $\omega_1 = 1$ AND $\omega_2 = 0$. $\{k_p^{(i)}, k_d^{(i)}, k_i^{(i)}\}$ ARE CONTROL GAINS.

EA	$k_p^{(1)}$	$k_p^{(2)}$	$k_d^{(1)}$	$k_d^{(2)}$	$k_i^{(1)}$
Known [26]	**3.63***e*+2	**3.58***e*+2	**6.79**	**1.77***e*+1	**3.54***e*+1
CMA-ES	**4.00***e*+2	**4.00***e*+2	**4.48***e*+1	**1.15***e*+1	**4.00***e*+2
OMNI	**3.99***e*+2	**3.99***e*+2	**4.47***e*+1	**1.17***e*+1	**3.99***e*+2
BUMDA	**3.99***e*+2	**3.99***e*+2	**2.13***e*+2	**8.27**	**3.84***e*+2
EA	$k_i^{(2)}$	$\mathcal{F}_1$	$\mathcal{F}_2$	$\mathcal{F}^*$	
Known [26]	**3.07***e*+2	**4.36***e*-3	**6.18***e*-3	**5.27***e*-3	
CMA-ES	**4.00***e*+2	**3.56***e*-3	**4.55***e*-3	**4.06***e*-3	
OMNI	**3.99***e*+2	**3.56***e*-3	**4.55***e*-3	**4.06***e*-3	
BUMDA	**3.99***e*+2	**3.11***e*-3	**5.04***e*-3	**4.07***e*-3	

2) Sequential vs Concurrent Optimization: This subsection is devoted to compare the sequential and concurrent methodologies in the complete problem of optimizing structure lengths and control parameters. The sequential methodology, for this case, is as follows: First, the static problem defined by Eq. 6 is solved, where a discretized circular region centered at the origin with radius $\mathbf{1}m$ is introduced as the reference workspace. Subsequently, the kinematic problem defined by Eq. 5 is used to obtain the joints position and velocity profiles. The desired end-effector trajectory is generated by taking 10 time-equidistant points from $\mathbf{X} = [(\mathbf{0.15}t/t_f + \mathbf{0.6})\cos(t/t_f\pi), (\mathbf{0.15}t/t_f + \mathbf{0.6})\sin(t/t_f\pi)]$, for $t_f = \mathbf{4}$ and $t \in [\mathbf{0}, \mathbf{4}]$. The points are interpolated using C^0-lines for the sake of observing the behavior of the control when tracking derivative-discontinuous trajectories. The last problem addressed is defined by Eq. 1 to find the control parameters of PID controllers for each joint. Table II, shows the best optimization parameters and objective function values found in 30 executions. The first row are results from the sequential optimization, columns 2 and 3 are from the static problem and the remaining from the dynamic. The second row is the concurrent optimization, which outperforms the best result obtained by the sequential methodology by $\mathbf{32.1434\%}$ and an average improvement of $\mathbf{32.1423\%}$ computed over the 30 executions. A graphical comparison of the sequential optimization with and without penalization is shown in Fig. 4. Torques have less drastic changes using $\boldsymbol{\lambda} = \mathbf{500}$ than with the unpenalized $\boldsymbol{\lambda} = \mathbf{0}$. Hence, it is notorious the effect of the penalization term weighted by $\boldsymbol{\lambda}$ in the proposed model of Eq. 1. As can be observed, smoothing the torques has an impact in the trajectory tracking. This problem has a relatively low dimensionality, hence, it is not the most challenging for the optimization algorithm, nevertheless the concurrent method delivers an evident better design. The explanation is that the

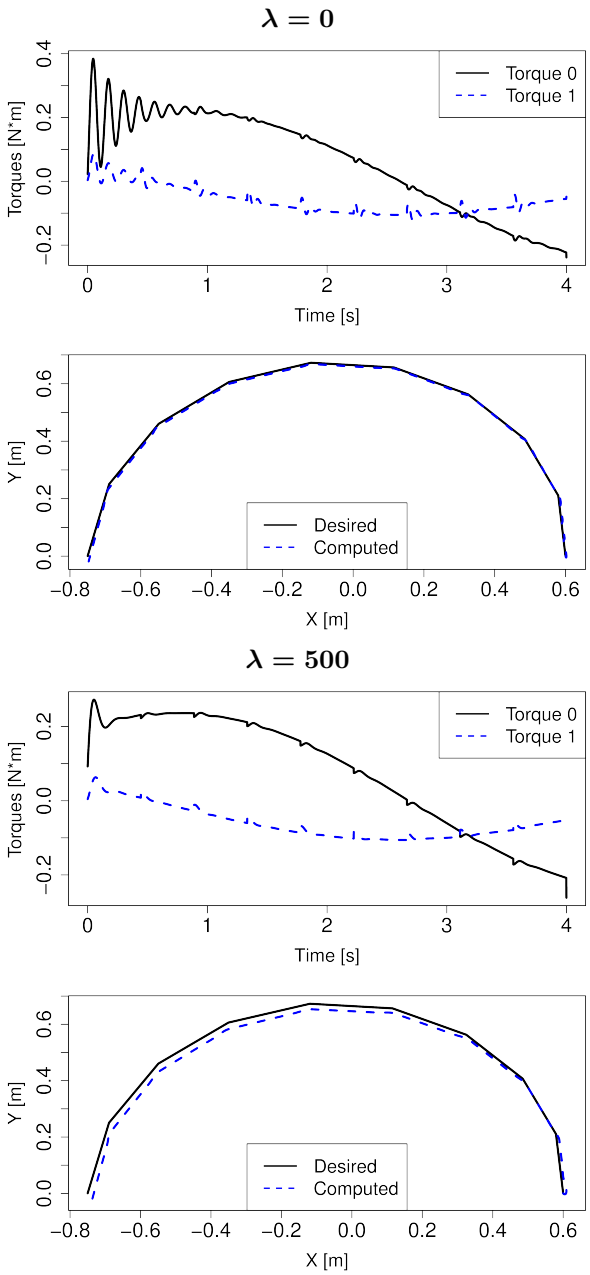

Fig. 4. Execution of the sequential methodology without ($\boldsymbol{\lambda} = \mathbf{0}$) and with ($\boldsymbol{\lambda} = \mathbf{500}$) penalization term in Eq. 1 for the two-link manipulator.

sequential methodology finds approximated optimal lengths, if such lengths are not optimal, for the dynamic model in Eq. 1, the whole design is suboptimal even if the optimum control gains are found for the given lengths. We argue that the sequential methodology could perform a better local search in each stage, but a wrong output from any stage, highly affects the whole design, while the concurrent methodology always performs a global search, thus it is more probable to find the global optimum during the whole optimization process. In the same vein, an increment in the problem dimensionality affects more the concurrent method than the sequential due to it only deals with subsets of optimization variables.

TABLE II
BEST SOLUTION FOR EQ. 1 WITH $\boldsymbol{\omega_1 = \omega_2 = 0.5}$ AND $\boldsymbol{\lambda = 0}$. LINK LENGTHS=$\{\boldsymbol{l_1}, \boldsymbol{l_2}\}$, CONTROL PARAMETERS=$\{\boldsymbol{k_p^{(i)}}, \boldsymbol{k_d^{(i)}}, \boldsymbol{k_i^{(i)}}\}$ AND OBJECTIVE VALUE=$\boldsymbol{\mathcal{F}}$. SEQUENTIAL=$\boldsymbol{S}$ AND CONCURRENT=$\boldsymbol{C}$.

Met.	$\boldsymbol{l_1}$	$\boldsymbol{l_2}$	$\boldsymbol{k_p^{(1)}}$	$\boldsymbol{k_p^{(2)}}$	
S	**6.6253***e*-1	**5.814***e*-1	**5.8936**	**2.6337**	
C	**4.2791***e*-1	**4.00***e*-1	**3.7093**	**1.8759**	
Met.	$\boldsymbol{k_d^{(1)}}$	$\boldsymbol{k_d^{(2)}}$	$\boldsymbol{k_i^{(1)}}$	$\boldsymbol{k_i^{(2)}}$	$\boldsymbol{\mathcal{F}}$
S	**1.1897***e*-1	**4.8364***e*-2	**1***e*-12	**4.5123***e*-10	**9.3273***e*-1
C	**4.78273***e*-2	**1.8013***e*-2	**1***e*-12	**1***e*-12	**6.3292***e*-1

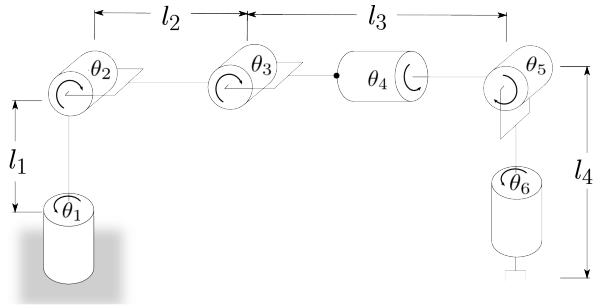

Fig. 5. Case study: Anthropomorphic serial manipulator. Scheme of its structure.

B. Case study: Anthropomorphic serial manipulator

The anthropomorphic 6-DOF manipulator is of general purpose and has great applicability for its manipulability in 3D. Popular robots like the PUMA, Motoman HP6, Fanuc 10L and ABB S4 2.8, are examples of this type of manipulator [27]. Fig. 5 shows a representation of the 6-DOF manipulator.

1) Sequential vs Concurrent Optimization: We present a comparison between the sequential and concurrent methodologies using the best performed optimizer from the above sections: CMA-ES. In the sequential methodology the process is as follows: A static optimization problem, defined by Eq. 6, is solved for a discretized spherical volume with center at the origin and radius $\boldsymbol{1m}$. It maximizes the workspace volume constrained to manipulability and joint limits. The discretized desired workspace is reconstructed with cubes. The number of reached cubes is taken as an approximation to the volume of the reachable workspace. Subsequently, a kinematic problem, defined by Eq. 5, is solved to obtain positions and velocity profiles for each joint by minimizing the tracking error with respect to the desired trajectory of the end-effector given by $\mathbf{X} = [(\mathbf{0.15}t/t_f + \mathbf{0.6})\cos(t/t_f\pi),\ (\mathbf{0.15}t/t_f + \mathbf{0.6})\sin(t/t_f\pi), \mathbf{0.80}t/t_f - \mathbf{0.2}, \beta, \gamma, \eta]$, where β, γ, η are the direction cosines of the corresponding point. $\boldsymbol{C^0}$-curves are used to interpolate the discretized trajectory for testing the effect of the penalization term weighted by $\boldsymbol{\lambda}$ in the final dynamic optimization process defined by Eq. 1, to optimize the PID control parameters of each joint. On the other hand, the concurrent methodology is applied to minimize the objective function of Eq. 1 using also the desired trajectory delivered by the kinematic optimization. Hence, both methodologies use the same desired trajectory. Table III shows the best parameters and objective function values from 30 executions of the sequential and concurrent methodologies. In this case, the sequential delivers a better result than the best of the concurrent method by $\mathbf{1.4384\%}$. Nevertheless, in both cases, in 20 from 30 executions the methods found geometry and control parameters that can be evaluated during the whole simulation time. In the rest of the executions, the resulting manipulator cannot track the whole trajectory due to singularities. A graphical comparison of the two methodologies is shown in Fig. 6; it presents the applied torque and the angular position error for each joint throughout time, and the trajectory tracking using the dynamic control for the best solution of the two methodologies. From the successful executions, the sequential delivers an average objective function value of 1.583717 and the concurrent of 2.766822, with a corresponding standard deviation of 0.0346445 and 0.5606268, respectively. Hence, the sequential methodology performs the best in this case. The higher variance in the concurrent design provides of empirical evidence about the search: the concurrent method looks for a solution in a higher number of design configurations. Furthermore, according to the error plot in Fig. 6, the concurrent method delivers a smaller error, nevertheless, the torques from both methods are in the same scale. The explanation is that according to Table III, the concurrent requires larger control gains. Considering that the concurrent delivers a smaller error, possibly, another execution or a local refinement of the control gains could deliver better results than the sequential, which very possible cannot find better control gains considering that the lengths are fixed. Thus, future work could explore memetic algorithms and hybridization of this two methods.

TABLE III
BEST SOLUTIONS FOR EQ. 1 WITH $\boldsymbol{\lambda = 500}$. LINK LENGTHS=$\boldsymbol{l_i}$, CONTROL PARAMETERS =$\{\boldsymbol{k_p^{(i)}}, \boldsymbol{k_d^{(i)}}, \boldsymbol{k_i^{(i)}}\}$ AND OBJECTIVE VALUE=$\boldsymbol{\mathcal{F}}$. SEQUENTIAL=$\boldsymbol{S}$, CONCURRENT=$\boldsymbol{C}$.

Met.	$\boldsymbol{l_1}$	$\boldsymbol{l_2}$	$\boldsymbol{l_3}$	$\boldsymbol{l_4}$	$\boldsymbol{k_p^{(1)}}$	$\boldsymbol{k_p^{(2)}}$	$\boldsymbol{k_p^{(3)}}$	$\boldsymbol{k_p^{(4)}}$
S	**9.04**e-3	**0.677**	**0.5012**	**0.4987**	**0.3692**	**6.639**	**3.057**	**1.406**
C	**0.4**	**0.8**	**0.4393**	**0.4**	**9.385**	**7.236**	**8.582**	**7.549**
	$\boldsymbol{k_p^{(5)}}$	$\boldsymbol{k_p^{(6)}}$	$\boldsymbol{k_d^{(1)}}$	$\boldsymbol{k_d^{(2)}}$	$\boldsymbol{k_d^{(3)}}$	$\boldsymbol{k_d^{(4)}}$	$\boldsymbol{k_d^{(5)}}$	$\boldsymbol{k_d^{(6)}}$
S	**2.72**	**0.255**	**0.0171**	**0.1542**	**0.0642**	**0.0036**	**0.024**	**9.29**e-4
C	**7.38**	**2.35**	**0.1214**	**0.1650**	**0.0713**	**0.0123**	**0.021**	**3.02**e-3
	$\boldsymbol{k_i^{(1)}}$	$\boldsymbol{k_i^{(2)}}$	$\boldsymbol{k_i^{(3)}}$	$\boldsymbol{k_i^{(4)}}$	$\boldsymbol{k_i^{(5)}}$	$\boldsymbol{k_i^{(6)}}$	$\boldsymbol{\mathcal{F}}$	
S	**3.01**	**9.931**	**9.147**	**9.346**	**2.331**	**3.44**	**1.527**	
C	**10**	**1.9**	**8.846**	**1.862**	**0.1982**	**6.674**	**1.549**	

In summary, the sequential method delivers the best overall result in 30 executions, however, this solution is near from the best delivered by the concurrent method. From the 30 executions, only 20 are successful for both methods, i.e., in 2/3 of the executions the methods found a manipulator capable of closely tracking the desired trajectory. In the simplest 2-DOF case, the concurrent method is the best and it always finds a

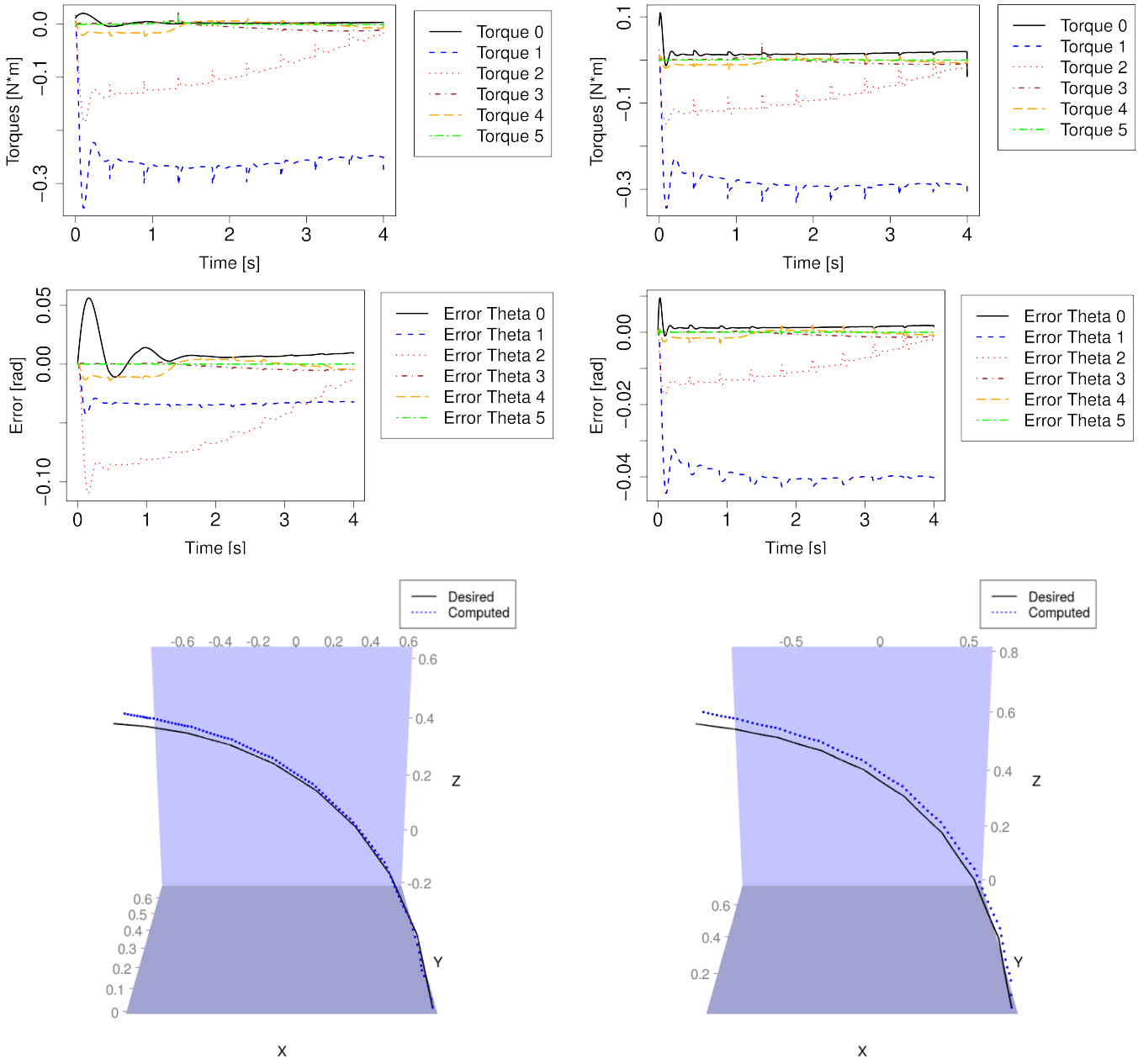

Fig. 6. Comparison of the sequential (left) and concurrent (right) methodologies for the anthropomorphic manipulator using the CMA-ES. First row: torques, second row: position errors and third row: executed and desired trajectories.

solution. Although the proposed optimization methodologies are performed offline, we would like to have results in an acceptable execution time. To give an idea, the concurrent methodology execution time is about 3 days and it is 61% of the sequential execution time.

2) Robustness of the design and parameter impact analysis: A correlation plot is presented in Fig. 7. It is a graphical representation of the statistical correlation matrix from the best solutions found by each methodology. These plots allow us to infer relationships between design parameters that affect the performance of the mechanism. The widest ellipses show uncorrelated variables or a low correlation, while tight ellipses with stronger color show a high correlation. Tight ellipses rotated clockwise in red color show positive-high correlation, in contrast, if they are rotated counter clockwise and in blue color, they show negative-high correlation. Thus, in Fig. 7 (top), at the first row, for the sequential methodology, we can infer that the objective value is closely related to some of the proportional and derivative control parameters. The last have a greater impact, mainly, $\boldsymbol{k_d^{(2)}}$, $\boldsymbol{k_d^{(3)}}$, since the correlation with the objective function is negative, which means that increasing $\boldsymbol{k_d^{(3)}}$ decreases the objective function. For the concurrent case, it is interesting that, clearly, $\boldsymbol{k_p^{(6)}}$ and $\boldsymbol{k_d^{(6)}}$ have the most impact to the objective function. When the correlation for

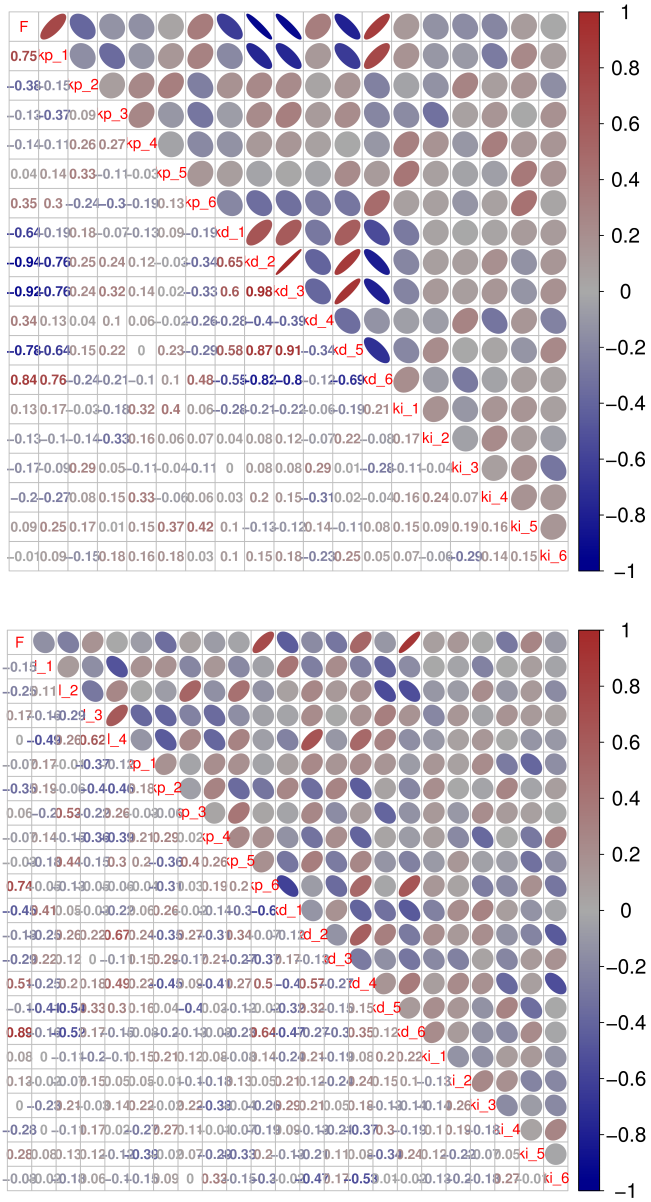

Fig. 7. Correlation analysis for the anthropomorphic case. Top: Sequential. Bottom: Concurrent.

some parameters is not clear, it means that modifying them in a closed vicinity does not impact the objective function significantly. This is a kind of robustness of the design. Standard deviations (SD) of the objective functions and control parameters give an idea of how much one can perturb the control parameters while maintaining a similar performance. For example, considering the sequential optimization, the SD of $\boldsymbol{k}_p^{(6)}$ is 3.19, and 0.03464 for the objective function. Hence, the small SD of the objective function, shows that, when a feasible design is found, it is also robust because it supports relatively large perturbations on the parameters without significantly changing the performance. This is expected if the algorithm converges to the same region of the search space.

Hence, our analysis shows that there are some parameters more important than others, in general some of the most important are the derivative control parameters. This analysis, also shows whether increasing a parameter value increases the objective function and vice versa. On one hand, the sequential method presents a lower variance than the concurrent, and a stronger correlation between the control parameters and the objective function, according to Fig. 7. This suggest that the solutions delivered by the sequential method highly depend on the last optimization step, while the solutions delivered by the concurrent method are not as dependent, as those of the sequential, on a few parameters. Notice that the sequential methods is searching for 18 optimum parameters in the last optimization step (6 proportional, 6 integral, and 6 derivative gains), while the concurrent is searching for 22 (the same than the sequential plus 4 lengths).

V. Conclusions

According to our experiments, we can give clear directions about the three metaheuristics tested (Omnioptimizer, BUMDA and CMA-ES), CMA-ES consistently delivers the best objective function values. Considering the sequential versus the concurrent optimization methodologies, the first one performs better for the anthropomorphic manipulator case, nevertheless, the concurrent method delivers similar results. In addition, the concurrent is best suited for a global search, the larger variance obtained in its results suggests that a more exhaustive exploration or a local search procedure inside this method could improve its performance, which is contemplated as future work.

Notice that in the sequential methodology for each set of lengths there is a different set of optimum control gains, because modifying the lengths, modifies the masses and manipulator dynamics. The optimization step that determines the lengths in the sequential method does not consider the dynamics, while the concurrent method does. Hence, the sequential method searches with high precision the control parameters for a given set of lengths and it is competitive if the lengths are an approximation close to optimal. In contrast, the concurrent method delivers a solution which considers the dynamics to set the lengths and control parameters using significantly less computational load. According to our results, the best solutions from both methodologies are not so far from each other in performance (objective function) but they are quite different in the parameter values. Our results suggest that the sequential method must be used if one is confident on the lengths approximation delivered by the first optimization step, while the concurrent method could be used to obtain a rough approximation of an optimum configuration which considers the dynamics to set simultaneously the lengths and the control parameters. Thus, future work considers to hybridize the two proposals, for example, by using the concurrent methodology to set the search limits of the optimization variables in the sequential method, or by using the steps of the sequential methodology to refine the solution given by the concurrent methodology. This paper gives directions for optimum manip-

ulator design using evolutionary algorithms, considering that the very same methodologies could be used in other scenarios, for instance, in reconfigurable robots, where the lengths and control parameters can be optimally set for each given task.

REFERENCES

[1] G.R. Zavala, A.J. Nebro, F. Luna, and C.A. Coello Coello. A survey of multi-objective metaheuristics applied to structural optimization. *Structural and Multidisciplinary Optimization*, 49(4):537–558, 2014.

[2] S. Fong, S. Deb, and A. Chaudhary. A review of metaheuristics in robotics. *Computers & Electrical Engineering*, 43(Supplement C):278 – 291, 2015.

[3] C. Lanni, S.F.P. Saramago, and M. Ceccarelli. Optimal design of 3r manipulators by using classical techniques and simulated annealing. *J. of the Brazilian Society of Mechanical Sciences*, 24(4):293–301, 2002.

[4] M. Ceccarelli and C. Lanni. A multi-objective optimum design of general 3r manipulators for prescribed workspace limits. *Mechanism and Machine Theory*, 39(2):119–132, 2004.

[5] Z. Du and H. Dong. Optimal dimension of redundant manipulator using the workspace density function. *Proc. of the Institution of Mechanical Engineers, Part C: J. of Mechanical Engineering Science*, 230(11):1787–1794, 2016.

[6] P.R. Bergamaschi, A.C. Nogueira, and S.F.P. Saramago. Design and optimization of 3r manipulators using the workspace features. *Applied Mathematics and Computation*, 172(1):439–463, 2006.

[7] G. Carbone, E. Ottaviano, and M. Ceccarelli. An optimum design procedure for both serial and parallel manipulators. *Proc. of the Institution of Mechanical Engineers, Part C: J. of Mechanical Engineering Science*, 221(7):829–843, 2007.

[8] S. Patel and T. Sobh. Task based synthesis of serial manipulators. *J. of Advanced Research*, 6(3):479–492, 2015.

[9] R.V. Rao. *Design Optimization of a Robot Manipulator Using TLBO and ETLBO Algorithms*, pages 163–169. Springer International Publishing, Cham, 2016.

[10] P.R. Bergamaschi, S.F.P. Saramago, and L.S. Coelho. Comparative study of sqp and metaheuristics for robotic manipulator design. *Applied Numerical Mathematics*, 58(9):1396–1412, 2008.

[11] S. Panda, D. Mishra, B.B. Biswal, and M. Tripathy. Revolute manipulator workspace optimization using a modified bacteria foraging algorithm: A comparative study. *Engineering Optimization*, 46(2):181–199, 2014.

[12] H. Al-Dois, A.K. Jha, and R.B. Mishra. Task-based design optimization of serial robot manipulators. *Engineering Optimization*, 45(6):647–658, 2013.

[13] T. Ravichandran, G.R. Heppler, and D.W.L. Wang. Task-based optimal manipulator/controller design using evolutionary algorithms. Technical report, University of Waterloo, 2004.

[14] G. Reynoso-Meza, J. Sanchis, X. Blasco, and M. Martínez. Algoritmos evolutivos y su empleo en el ajuste de controladores del tipo pid: Estado actual y perspectivas. *Revista Iberoamericana de Automática e Informática Industrial RIAI*, 10(3):251–268, 2013.

[15] B.K. Rout and R.K. Mittal. Optimal design of manipulator parameter using evolutionary optimization techniques. *Robotica*, 28(3):381–395, 2010.

[16] Y. Xia and J. Wang. A dual neural network for kinematic control of redundant robot manipulators. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 31(1):147–154, 2001.

[17] R.R. Santos, V. Steffen, and S.F.P. Saramago. Optimal task placement of a serial robot manipulator for manipulability and mechanical power optimization. *Intelligent Information Management*, 2(9):512–525, 2010.

[18] M. Pellicciari, G. Berselli, F. Leali, and A. Vergnano. A method for reducing the energy consumption of pick-and-place industrial robots. *Mechatronics*, 23(3):326–334, 2013.

[19] T. Ravichandran, D. Wang, and G. Heppler. Simultaneous plant-controller design optimization of a two-link planar manipulator. *Mechatronics*, 16(3):233–242, 2006.

[20] H.V.H. Ayala and L.S. Coelho. Tuning of pid controller based on a multiobjective genetic algorithm applied to a robotic manipulator. *Expert Systems with Applications*, 39(10):8968–8974, 2012.

[21] S. Hwang, H. Kim, Y. Choi, K. Shin, and C. Han. Design optimization method for 7 dof robot manipulator using performance indices. *Int. J. of Precision Engineering and Manufacturing*, 18(3):293–299, 2017.

[22] N. Hansen, A.S.P. Niederberger, L. Guzzella, and P. Koumoutsakos. A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. *IEEE Trans. on Evolutionary Computation*, 13(1):180–197, 2009.

[23] A. Klanac and J. Jelovica. A concept of omni-optimization for ship structural design. *Advancements in Marine Structures, Proceedings of MARSTRUCT*, pages 473–481, 2017.

[24] S.I. Valdez, A. Hernández, and S. Botello. A boltzmann based estimation of distribution algorithm. *Information Sciences*, 236:126–137, 2013.

[25] S. Botello-Aceves. *Concurrent Design Optimization of Kinematically Complex Mechanisms*. MSc dissertation, Center for Research in Mathematics, 2016.

[26] R. Sharma, K.P.S. Rana, and V. Kumar. Comparative study of controller optimization techniques for a robotic manipulator. In *Proc. of Int. Conf. on Soft Computing for Problem Solving*, pages 379–393. Springer, 2014.

[27] P. Corke. *Robotics, Vision and Control*, volume 73. Springer-Verlag, 2013.