MULTI-OBJECTIVE PREDICTIVE CONTROL: A SOLUTION USING METAHEURISTICS

Halim Merabti[1, 2] and Khaled Belarbi[3]

[1]Welding and NDT Research Centre (CSC).BP 64 Cheraga-Algeria.
[2]Dept of electronics, Faculty of technology, University of Constantine 1, Constantine, Algeria.
[3]Ecole polytechnique de Constantine, Campus Université de Constantine 3, Nouvelle Ville Ali Mendjeli, Constantine, Alegria

ABSTRACT

The application of multi objective model predictive control approaches is significantly limited with computation time associated with optimization algorithms. Metaheuristics are general purpose heuristics that have been successfully used in solving difficult optimization problems in a reasonable computation time. In this work , we use and compare two multi objective metaheuristics, Multi-Objective Particle swarm Optimization, MOPSO, and Multi-Objective Gravitational Search Algorithm, MOGSA, to generate a set of approximately Pareto-optimal solutions in a single run. Two examples are studied, a nonlinear system consisting of two mobile robots tracking trajectories and avoiding obstacles and a linear multi variable system. The computation times and the quality of the solution in terms of the smoothness of the control signals and precision of tracking show that MOPSO can be an alternative for real time applications.

KEYWORDS

Model predictive control, Metaheuristcis, Multiobjective Optimization.

1. INTRODUCTION

Model based predictive control (MBPC) is a form of control in which the current control action is obtained by solving on-line, at each sampling instant, a finite horizon open-loop optimal control problem, using the current state of the plant as the initial state and a model for predicting the future behavior of the plant [1]. The optimization yields an optimal control sequence and the first control in this sequence is applied to the plant . Usually, this strategy involves a single objective function to be minimized. Lately, however, multi-objective MBPC has began to attract interest [2][3][4][5]. For example, In [2], the authors use multi objective optimization to tune non linear model predictive controllers basing on a weighted sum of objective functions and in [3] authors showed that it is possible to compute Pareto optimal solution as an explicit piecewise affine function after recasting the optimization problem associated with the multi objective MPC as a multi parametric multi objective linear or quadratic program. However, application of multi objective model predictive control approaches is significantly limited with computation time associated with optimization algorithms. On the other hand, Metaheuristics such as genetic algorithms [6], particle swarm optimization [7], and gravitational search algorithm [8] are general purpose heuristics which have been successful in solving difficult optimization problems in a reasonable computation time. Basically, most metaheuristcs have been extended to multiobjective optimization, we thus find for instance the non dominated sorting genetic algorithm II (NSGA-II) [9], non-dominated sorting particle swarm optimizer for multi objective optimization (NSPSO) [10], multi objective particle swarm optimization MOPSO [11] [12], multi objective gravitational search algorithm (MOGSA) [13].

Metaheuristics may be used in many ways in the multiple objective context. In this work we use multi-objective metaheuristics to generate a set of approximately Pareto-optimal solutions in a single run. We present a comparison between two different multi objective metaheuristics, Multi Objective Particle Swarm Optimization, MOPSO, and Multi Objective Gravitational Search Algorithm, MOGSA, for the solution of the multi objective optimization problem arising in MOMPC.

The paper has the following structure: Section 2 covers the multi objective optimization concepts, section3 gives the formulation of the Multi objective non linear model predictive control, section4 provides the description of the MOPSO and MOGSA and section5 describes the applications of the above metaheuristics for the control of linear and non linear systems.

2. MULTI OBJECTIVE OPTIMIZATION

The problem of multi objective optimization has the following general form:

$$min_{x\in\Omega}\{\mathrm{F}(x)\} \tag{1}$$

Ω:is the decision variable space, F: objective functions vector

x : vector of decision variables from the decision variable space

$$\mathrm{F}:\Omega \rightarrow R^k, \mathrm{F}(x) = [f_1(x), f_2(x), \dots, f_k(x)]^T \tag{2}$$

With

$$\begin{cases} g_i(x) \le 0 & i = 1,2,\dots,m \\ h_j(x) = 0 & j = 1,2,\dots,p \end{cases} \tag{3}$$

And k is the number of the objective functions, $f_i: R^n \rightarrow R$, with $x = [x_1, x_2, \dots, x_n]^T$. The solution of this problem is noted $x^* = [x_1^*, x_2^*, \dots, x_n^*]^T$

Definition1: Given $x,\ y \in \Omega$, x is said to *dominate* y, or $x \prec y$ if and only if:
$f_i(x) \le f_i(y)$ *i=1…n* and $\mathrm{F}(x) \ne \mathrm{F}(y)$,

Definition2: $x^* \in \Omega$ is a *Pareto optimal* if and only if there does not exist another decision vector $y \in \Omega$ such that $y \prec x^*$

Definition3: Set of Pareto optimal solutions is defined as:

$$PF = \{x \in \Omega | x\ is\ a\ Pareto\ optimal\ soltion\}$$

3. FORMULATION OF THE MULTI OBJECTIVE NON LINEAR MODEL PREDICTIVE CONTROL

Consider a non linear system described by the discrete state space model:

$$x(k+1) = f\big(x(k), u(k)\big) \tag{4}$$

Where *x(k)* is the state , *u(k)* the control signal and f is a continuous mapping.

The control signal $u(k)$ is such that :

$$u(k) \in U \subset \mathbb{R}^m \tag{5}$$

U is a compact convex set with $0 \in \text{int}(U)$ and $f(0,0) = 0$. The state may be constrained to stay into a convex and closed set: $x(k) \in \mathbb{X}$

The problem is to regulate the state to the origin by solving the following optimization problem:

$$min_{\mathcal{U}} J_N(x, k, U) \tag{6}$$

Where

$$J_N(x, k, U) = F\big(x(k + N)\big) + \sum_{i=k}^{k+N-1} L\big(x(i), u(i)\big) \tag{7}$$

In multi objective model predictive control, we consider the optimization of criteria using multi objective optimization approaches:

$$min_{\mathcal{U}} \quad [J_0(U, x), J_1(U, x), \dots, J_l(U, x)]' : R^S \times R^n \to R^{l+1} \tag{8}$$

Subject to:

$$x(k + 1) = f\big(x(k), u(k)\big) \tag{9}$$

$$x(k) \in \mathbb{X}, k = 1, \dots, N \tag{10}$$

$$u(k) \in U, k = 0, \dots, N - 1 \tag{11}$$

$$x(k + N) \in \Omega \tag{12}$$

where:

$s = N * m$, $U[u'_0, \dots, u'_{N-1}]$ is the sequence of future control moves to be optimized.

The problem of multi objective model predictive control is to minimize, at each sampling time, the l following functions cost:

$$J_i(U, x) = F_i\big(x(k + N)\big) + \sum_{k=j}^{k+N-1} L_i(x(j), u(j)) \quad with\, i = 0, \dots, l \tag{13}$$

$F_i\big(x(k + N)\big)$ is a weight on the final state. Moreover, the final state may be constrained to be in a final region $x(k + N) \in X_f \subset X$

The weight F_i and the final region are introduced to guarantee stability of the non linear MPC.

The solution gives the set of Pareto front and only one Pareto optimal solution is selected and applied at sampling instant k. The procedure is repeated at each sampling time.

4. META HEURISTICS FOR MULTI OBJECTIVE OPTIMIZATION

The ultimate objective in multi objective optimization is the identification of the Pareto optimal which contains the non dominated solutions. In the last decade, researches were oriented towards extending metaheuristic such as particle swarm optimization, PSO, ant colony optimization, ACO, gravitational search algorithm, GSA, and other, to the solution of multi objective optimization problems. In this section, we introduce the two metaheuristics studied in this work.

4.1. The MOPSO:

Particle swarm optimization is an evolutionary computation technique developed by Kennedy and Eberhart in 1995 [7]. The particle swarm concept originated as a simulation of a simplified social system. The success of the particle swarm optimization algorithm motivated the researchers to apply it to multi-objective optimization problems. The MOPSO [10] is one of the algorithms proposed to solve the multi objective optimization problem using particle swarm optimization algorithm. The MOPSO maintains two archives, one for storing the globally non-dominated solutions and the other for storing the individual best solutions attained by each particle. Basically, the updating of the particle is performed as follows:

$$V(t+1) = w * V(t) + R_1 * \left(p_{best}(t) - p(t)\right) + R_2 * (Rep(h) - p(t)) \tag{14}$$

$$p(t+1) = p(t) + V(t+1) \tag{15}$$

Where V is the particle velocity, *p(t)* is the curent position of the particle, w is a constant , R_1 and R_2 are random numbers in [0 1]. *REP* is a repository where are stored the positions of the non dominated particles and *h* is an index in the repository that is introduced to ensure some fitness sharing [10]. $P_{best}(t)$ is the best solution found by the particle. *REP* is updated by inserting the currently nondominated positions and dominated positions are eliminated. The size of the repository being limited when it becomes full and particles in less populated areas are given priority over those highly populated regions.

4.2. The MOGSA:

In the gravitational search algorithm, GSA, [8] objects attract each other by the force of gravity which causes a global movement of all objects towards the objects with heavier masses. Hence, masses cooperate using a direct form of communication, through gravitational force. The heavy masses, corresponding to good solutions, move more slowly than lighter ones. In GSA, each mass has four specifications: position, inertial mass, active gravitational mass, and passive gravitational mass. The position of the mass corresponds to a solution of the problem, and its gravitational and inertial masses are determined using a fitness function. The algorithm is evolves by properly adjusting the gravitational and inertia masses. Eventually, masses will be attracted by the heaviest mass which presents an optimum solution in the search space.

More specifically, if there are N objects in the solution set with positions x_i, *i=1…N.* At iteration *t*, the mass of each object is given by:

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{16}$$

with

$$m_i(t) = \frac{fitness_i(t) - wst(t)}{bst(t) - wst(t)} \tag{17}$$

$fitness_i(t)$ is the value of the cost function (fitness) of object *i* while *bst(t)* and *wst(t)* are respectively the best and worst values of fitness of all objects. Let *Kbest* be the set of *k* objects with the best fitness values, then the total force acting on a given object originates from heavier objects and is given by:

$$F_i(t) = \sum_{j \in Kbest} rand_j \, G(t) \frac{M_j(t) M_i(t)}{D_{ij}(t) + \varepsilon} (x_j(t) - x_i(t)) \tag{18}$$

with $rand_j \in [0,1]$ is a random number, *G(t)* is the gravitational constant, $D_{ij}(t)$ is the Euclidean distance between objects *i* and *j* and a small number. The gravitational constant given by:

$$G(t) = G_0 e^{-\alpha t / tmax} \tag{19}$$

G_0 is the initial value, α is a constant, *tmax* the maximum number of iterations.
The acceleration of object *i* at iteration *t* is then given by:

$$a_i = \frac{F_i(t)}{M_i(t)} \tag{20}$$

Position $x_i(t)$ and velocity $v_i(t)$ of the object at iteration t are given by:

$$v_i(t) = rand_i v_i(t) + a_i(t) \tag{21}$$

$$x_i(t) = x_i(t) + v_i(t+1) \tag{22}$$

The MOGSA presented in[13] is a multi objective optimization algorithm which classifies the population into different Pareto fronts, ranking individuals by using the Pareto front and the crowding distance concept from the NSGA-II. In order to calculate the multi-objective cost (fitness), it applies a linear bias *br* to the *r*th ranked element by using the expression: br =1/r, obtaining values from 1 to 1/N. Thus, a sorted population with a single fitness value is obtained.

5. APPLICATION

5.1. Example01

Consider four robots each of which has two wheels, two real and two virtual. The real robots track the virtual ones. It is assumed that there is a pure rolling. The kinematic model of the real robots is given by:

$$\dot{x}_i(t) = \frac{v_{ri}(t)+v_{li}(t)}{2} \cos\theta_i(t) \tag{23}$$

$$\dot{y}_i(t) = \frac{v_{ri}(t)+v_{li}(t)}{2} \sin\theta_i(t) \tag{24}$$

$$\dot{\theta}_i(t) = \frac{v_{ri}(t)-v_{li}(t)}{b} \tag{25}$$

Where $i=1, 2$, $v_{ri} \in R$ and $v_{li} \in R$ are t he right and left linear velocities of the wheels of the real robot i, $b \in R$ is the distance between the wheel centers. θ_i is the robot orientation and ω_i are the angular velocities.

The objective is to find a control law defined by $v_{ri}(t)$, $v_{li}(t)$ ($i=1, 2$) that allows the robots to:

- track given reference trajectories defined by:$[x_{ri}(t)\ y_{ri}(t)]$, $i = 1,2$, respectively
- Avoid fixed obstacles on the trajectories
- Avoid collision between them.

This problem is set as a multi-objective model predictive control with constraints that will solved using the above meta heuristics, MOPSO and MOGSA. Obstacle avoidance is ensured by adding a constraint to the MBPC problem: if the distance between the obstacle and the robot is less than a given value r then a penalty is added to the cost function. The same idea is used to avoid the collision between robots. The first reference trajectory is given by:

$x_{r1}(t) = \cos(\omega_0 t)$; $y_{r1}(t) = \sin(2 * \omega_0 t)$; ω_0=0.02 rad/s is the signal pulsation, the second reference trajectory is given by: $x_{r2}(t) = \cos(\omega_0 t + \varphi)$; $y_{r2}(t) = \sin(2 * \omega_0 t + \varphi)$;
The control signals are constrained to: $-0.7(m/s) \le v_{ri} \le 0.7\ (m/s)$; and
$-0.7(m/s) \le v_{li} \le 0.7(\ m/s)$. The sampling time is T=0.1second.

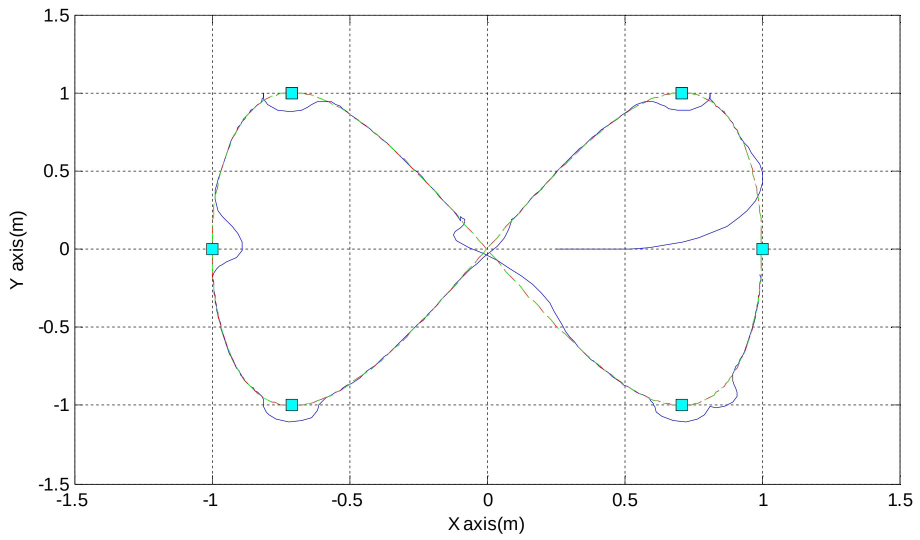

Figure 1. ROBOT1 trajectory

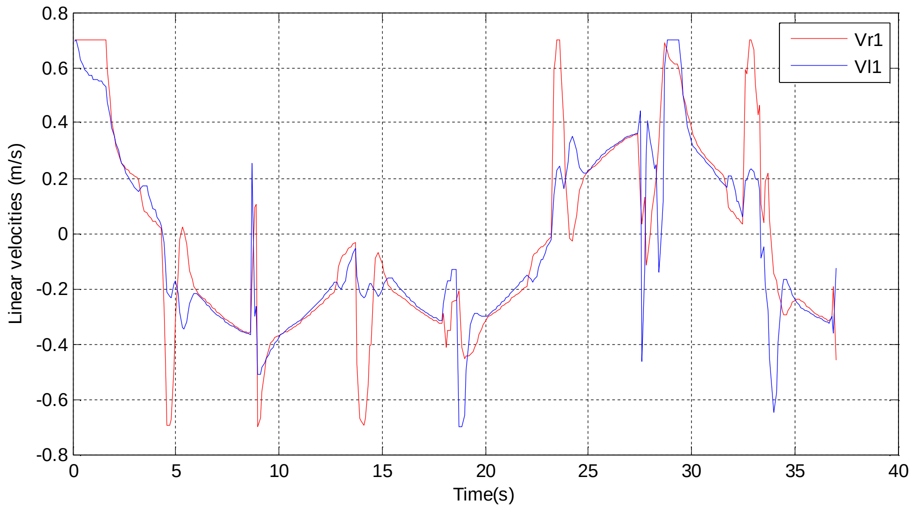

Figure 2. Control signals for ROBOT1

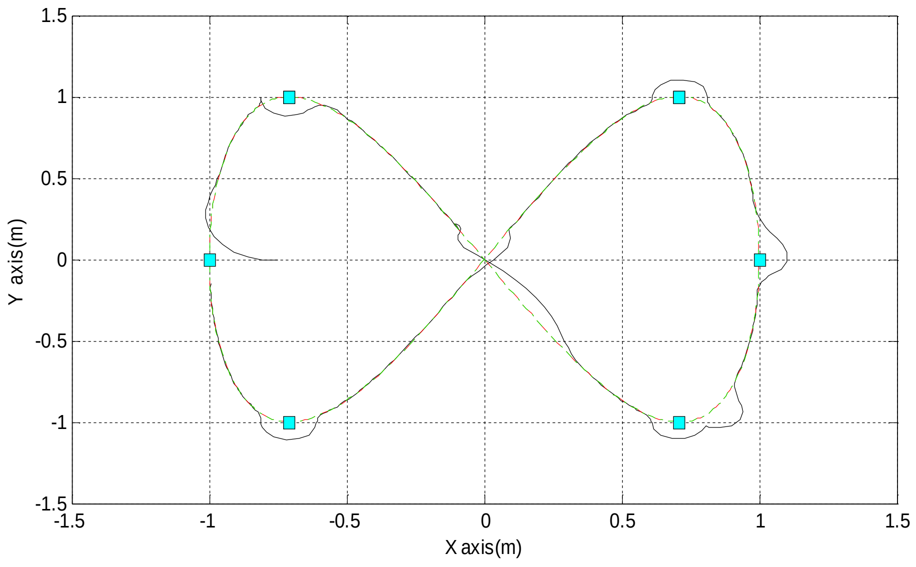

Figure 3. ROBOT2 trajectory

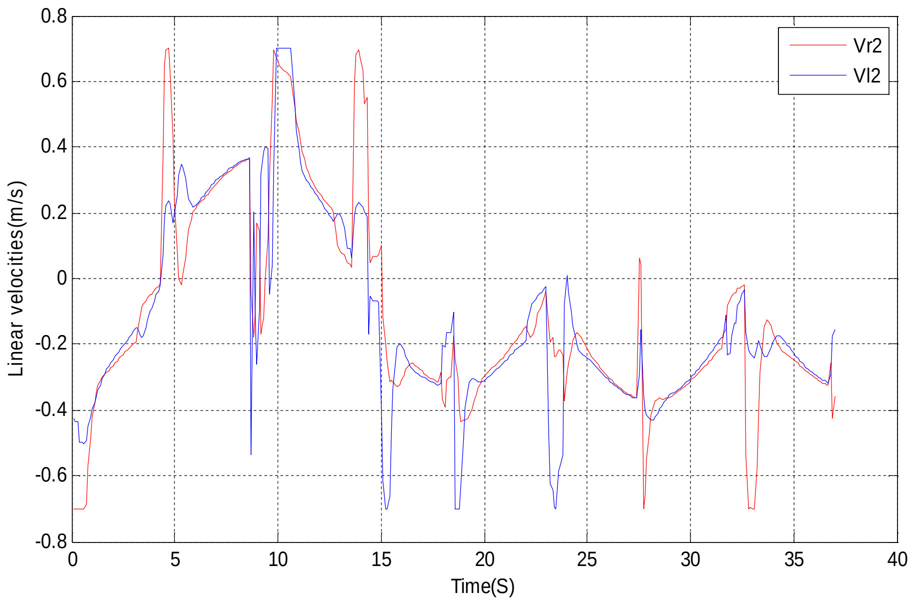

Figure 4 . Control signals for ROBOT2

The two algorithms are run until a satisfactory response is obtained. The collision point of the robots is *m(0,0)*. Robots start from their initial positions (0.25, 0), (-0.75, 0) and track their own trajectories, each one avoids the first fixed obstacle and continues its traveling to the collision point. It is observed that the second robot continues its tracking and the first avoids it by decreasing its velocities to keep a good safe distance. Then, robots continue their tracking and avoid fixed obstacles encountered.

Computation times are given in table 1, where it can be seen that MOPSO-NMPC out performs the other algorithm MOGSA-NMPC which produce a similar tracking and avoiding but with a longer run time. The constraints on the control signals are always satisfied as shown in figure (2) and (4). All the computation times are performed on *Intel® I3, 2120 CPU 3.30 GHz with 4Go RAM.*

Table 1. Computation time for the first example

	MOPSO-NMPC	MOGSA-NMPC
Computation time	≤ 9ms	≤ 190ms

5.2. Example02

In this second example we consider a multivariable system taken from [2]:

$$x(k+1) = Ax(k) + Bu(k)$$

$$A = \begin{bmatrix} 1.3433 & 0.1282 & 0 \\ -0.1282 & 1.2151 & 0 \\ 0.1282 & 0.0063 & 1.2214 \end{bmatrix};\ B = \begin{bmatrix} 0.1164 & 0.0059 \\ -0.0059 & 0.1105 \\ 0.1166 & -0.1105 \end{bmatrix} \tag{26}$$

$x(k) \in R^3,\ u(k) \in R^2.$

As given in [2] the problem consists in minimizing three quadratic performance indexes ($i=3$);

$$\min_u \left\{ J_i(U, x(t)) = x^T_{t+\frac{N}{t}} P_i x_{t+\frac{N}{t}} + \sum_{k=0}^{N-1} x^T_{t+\frac{k}{t}} Q_i x_{t+\frac{k}{t}} + u^T_{t+k} R_i u_{t+k} \right\} \tag{27}$$

In order to ensure stability, the final cost matrix P_i is calculated from the algebraic Riccati equation with the assumption that the constraints are not active for $k \geq N$. The weight matrices are [2]:

$$Q1 = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}; Q2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}; Q3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{28}$$

$$R1 = R2 = R3 = I2$$

$$x(0) = [2 \quad 2 \quad -1]'$$

The task is to regulate the system to the origin. To this aim, we design a controller based on the multi objective optimization problem, using MOPSO then MOGSA.

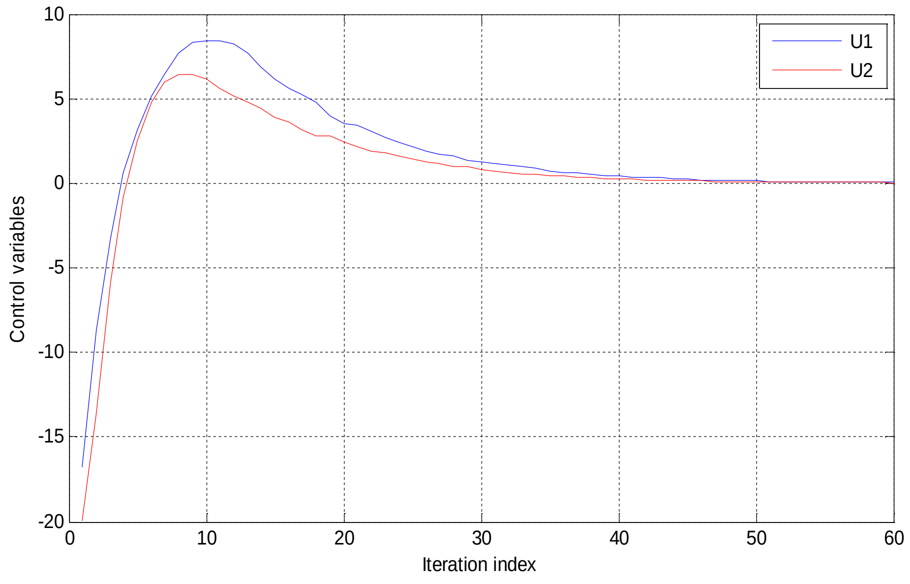

Figure 5. Control signals for example2, MOPSO and MOGSA

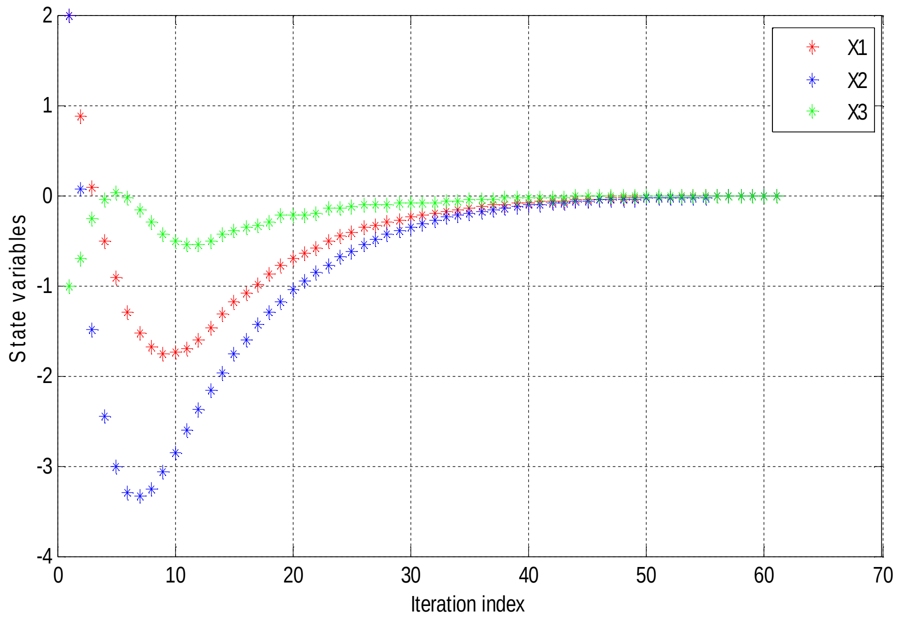

Figure 6. State variables for Example 2

It can be seen, from the results of the simulation shown in figure 5 and figure 6, that the control signals and the states are quite similar for both algorithms. The system is stabilized in about 40 iterations while satisfying the constraints. Again, these results are very similar to those obtained in [2] using a receding horizon approach to the multi objective control problem. Table 2 gives the computational time where one can see that the MOPSO algorithm is the fastest.

Table 2. Computation time for the second example

	MOPSO-MPC	MOGSA-MPC
Computation time	$\leq$ 4ms	$\leq$ 200ms

5. CONCLUSION

In this work, we compared the use of two multi objective metaheuristics, MOPSO and MOGSA, to generate a set of approximately Pareto-optimal solutions in a single run. Two examples were studied, a nonlinear system consisting of two mobile robots tracking trajectories and avoiding obstacles and a linear multi variable system. The computation times and the quality of the solution in terms of the smoothness of the control signals and precision of tracking show that MOPSO can be an alternative for real time applications.

REFERENCES

[1] J.M.Maciejowski, ’Predictive Control with Constraints’, United States Prentice Hall, 2001.
[2] D.D.Vito and S.Riccardo, ‘A receding horizon approach to the multiobjective’,in Proceedings of the 46th IEEE Conference on Decision and Control, New Orleans, LA, USA, Dec. 12-14, 2007
[3] A.Bemporad and D. Muñoz de la Peña,‘ Multiobjective model predictive control‘, Automatica, vol. 45, pp. 2823-2830, 2009.
[4] J.Hu, J.Zhu, G.Lei, G. Platt, and D.G.Dorrell, ‘Multi-Objective Model-Predictive Control for High-Power Converters’, IEEE Transactions on energy conversion, vol. 28, NO. 3, pp. 652 – 663, 2013.
[5] H. Hana, H. Qiana, J. Qiao, 2014, ‘Nonlinear multiobjective model-predictive control scheme for wastewater treatment process’ , Journal of Process Control, vol. 24, pp. 47–59. 2014.
[6] D.E.Goldberg, ‘Genetic Algorithms in Search’, Optimization and Machine Learning, Addison-Wesley, 1989.
[7] R.C.Eberhart, Y.Shi, ‘Particle swarm optimization: developments, applications and resources’. Proc. Congress on Evolutionary Computation. Seoul, Korea. 2001.
[8] E.Rashedi, H.Nezamabadi-pour, S.Saryazdi, ‘A Gravitational Search Algorithm. Information Sciences’: 179: 2232–2248. 2009
[9] K.Deb, A.Pratap, S.Agarwal, T.Meyarivan, ’A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II’, IEEE Transactions on evolutionary computation, vol. 6, NO. 2. 2002
[10] X.Li. A Non-dominated Sorting Particle Swarm Optimizer for Multiobjective Optimization. In Genetic and Evolutionary Computation - GECCO 2003, volume 2723 of LNCS, pp. 37–48, 2003.
[11] C.Coello, M.Lechuga, ’A Proposal for Multiple Objective Particle Swarm Optimization’. In Proceedings of the Congress on Evolutionary Computation, part of the 2002 IEEE World Congress on Computational Intelligence, Hawaii, pp. 1051–1056. 2002.
[12] C.Coello, G.T. Pulido, M.S. Lechuga,. ‘Handling Multiple Objectives With Particle Swarm Optimization’. IEEE Transactions on Evolutionary Computation, vol 8, pp. 256–279. 2004.
[13] D.L.Gonzalez-Alvarez, M.A.Vega-Rodrıguez, J.A.Gomez-Pulido, J.M. Sanchez-Perez, ’Applying a multiobjective gravitational search algorithm (MO-GSA) to discover motifs’. In International Work Conference on Artificial Neural Networks (IWANN’11), Lecture Notes in Computer Science, vol. 6692, pp. 372–379. 2011.