THE ANNALS OF "DUNĂREA DE JOS" UNIVERSITY OF GALATI
FASCICLE III, 2017, VOL. 40, NO. 2, ISSN 2344-4738, ISSN-L 1221-454X
ELECTROTECHNICS, ELECTRONICS, AUTOMATIC CONTROL, INFORMATICS

REAL-TIME OPTIMAL CONTROL USING PARTICLE SWARM OPTIMIZATION

Viorel MINZU

Department of Automation and Electrical Engineering, "Dunarea de Jos" University of Galati, Romania

Abstract: Optimal Control Problems can be solved by a well known metaheuristic, Particle Swarm Optimization. A version of this metaheuristic has been used to develop an algorithm that solves such a problem. The solution can be used to control the considered dynamic system only in open loop. Therefore, the main problem is how to use the PSO algorithm in a control structure that works in closed loop. The solution of this problem is to use the well known control structure Model Predictive Control whose controller has an optimization algorithm that minimizes the prediction error. The proposal of this work is to confer this role to the PSO based algorithm because it would minimize implicitly the prediction error.

Keywords— optimal control, hybrid topology particle swarm optimization, Model Predictive Control

1. INTRODUCTION

Particle Swarm Optimization (PSO) has very good abilities to solve many types of optimization problems. In a previous paper [18], the author proposed a PSO based algorithm dealing with the Optimal Control Problems (OCP). This one was tested upon a very simple OCP, but whose optimal solution can be expressed theoretically and compared with the solution of the PSO based algorithm.

This paper is the continuation of the work presented in [18], because the solution proposed here can be used only in an open loop control system. The solution is a quasi-optimal sequence of control input values that can be used to control the considered dynamic system. Therefore, the main problem is how to use the proposed PSO algorithm in a control structure that works in closed loop and takes into account the feedback information coming from the real process.

The section II is devoted to the presentation of the PSO (see [7]). This is a well-known metaheuristic that has proved with success its ability to solve many types of optimization problems as in [17], [13], [14]. The problems may have continuous, discrete or even binary models as in [12]. The success of PSO is due to the great theoretical support concerning their convergence, stability and complexity as in [1], [6], [8], [15].

The algorithm proposed in [18] is based on Hybrid Topology Particle Swarm Optimization (HTPSO), which is an improved version of PSO metaheuristic (see for example [12]) having better communication abilities among particles. This HTPSO algorithm devoted to solving an OCP is used in this work to implement the closed loop structure as well.

There is an important preoccupation with solving an OCP by metaheuristics, because of their big complexity. Details concerning the implementation of such an algorithm are given in [10] and [16]. In many useful applications, a dynamic system is modeled by a set of ordinary differential and algebraic equations, but also by a set of constraints (see [10], [11]).

In Section II, a brief description of HTPSO metaheuristic is also given. In this version of PSO, the local topology involves a direct communication between particles. Only the equations that are essential for an implementation are specified.

The section III is a general aperçu of how an OCP can be modeled. The differential and algebraic constraints that define the dynamic system behavior are briefly given.

Only some details concerning the implementation of HTPSO based algorithm are given in Section IV. The solution coding, the *step control* technique, the particles' position limitation and other particularities of the algorithm are described in this section. The reader of this paper can find also important detail in [18].

A case study is presented in section V, namely an optimal control problem for a DC motor with bilocal constraints. We have not chosen this example of OCP because it is less complex, but because we can find out its theoretical solution. So, a comparison with the solutions of the closed loop structure can be done. The results, obtained with HTPSO based algorithm, are analyzed in order to evaluate its efficiency. In this stage, we have the version of the algorithm used eventually in the open loop structure.

The Section VI presents a control structure using Model Predictive Control, which firstly introduces the closed loop structure that is crucial for a real-time application development. Secondly, the optimization algorithm that is within the controller and minimizes the prediction error will be replaced by the HTPSO based algorithm. It can be shown that the HTPSO based algorithm makes implicitly the minimization of the prediction error.

The efficiency of the MPC having the HTPSO based algorithm as a part of its controller is analyzed in section VII. Two simulation series have been made that have proved the very good behavior of the implemented control structures. Some conclusions are drawn in Section VIII that emphasis the efficiency of the control structure using HTPSO based algorithm.

2. SOME ELEMENTS CONCERNING THE HTPSO

Particle Swarm Optimization (PSO) is a population based metaheuristic combining basically two strategies, *gradient based search* with *hill climbing*.

In the incipient phase of the swarm's evolution, every particle explores new regions of the search space. As the searching process goes along, the exploration diminishes and the exploitation is strengthened and the particles converge to the best solution of the optimization problem (OP). The main aspects concerning the analysis and the design of PSO algorithms are presented in [17]. The convergence of PSO algorithms is treated in paper [6] that provides also a guide for the algorithm's parameters choice.

The swarm is modeled by a number of N particles, every particle being represented through 3 components vector [3], classically denoted by $\left(\vec{X}_i, \vec{V}_i, \vec{P}_{best_i}\right)$. Every component of a particle is a m-dimensional vector representing the position, the speed and the best personal position reached in the searching process (the best personal experience), respectively (see [5]). It holds:

$$(1)\ \vec{X}_i = \left(x_i^1, x_i^2, \ldots, x_i^d, \ldots, x_i^m\right),\ i = 1, 2, \ldots, N$$

$$(2)\ \vec{V}_i = \left(v_i^1, v_i^2, \ldots, v_i^d, \ldots, v_i^m\right),\ i = 1, 2, \ldots, N$$

$$(3)\ \vec{P}_{best_i} = \left(p_{best_i}^1, \ldots, p_{best_i}^d \ldots, p_{best_i}^m\right),\ i = 1, 2, \ldots, N$$

The evolution of the particle swarm is achieved through an iterative process. Let s be the step of this process, $1 \le s \le T$, where T is the estimated number of steps until convergence.

Let $\vec{P}_{gbest}$ be the best position reached by a particle of the swarm until the current step, called "global best" position:

$$(4)\ \vec{P}_{gbest} = \left(p_{gbest}^1, \ldots, p_{gbest}^d \ldots, p_{gbest}^m\right)$$

At the end of the algorithm, i.e. when convergence is reached, this is the OP's solution. The essential for the PSO algorithm is the updating of the particle speed and position, at every step of the searching process.

In our work we have used an improved version of PSO, namely Hybrid Topology Particle Swarm Optimization (HTPSO) that define a *local topology* of the swarm, regarded as a communication network. The local topology implies the existence, for any particle, of a "social neighborhood", i.e. a set of 3-5 particles that inform the particle #i about their best personal experience. These neighborhoods are decided in a deterministic or random way, at every step of the algorithm (see [4]). The algorithm determines the local best position, $\vec{P}_{lbesti}$, as being the best personal experience of the particles belonging to the "social neighborhood" and the particle #*i* itself. It holds:

$$(5)\ \vec{P}_{lbesti} = \left(p^1_{lbest_i}, \ldots, p^d_{lbest_i} \ldots, p^m_{lbest_i}\right), d = 1, \cdots, m;$$

Hence, when we use the HTPSO algorithm, the speed and the position updating, for the particle #*i*, is done using the following equations:

$$(6)\ v_i^d(s+1) = w \times v_i^d(s) + C_1 \times rand_1 \times \left(p^d_{best_i}(s) - x_i^d(s)\right) +$$

$$C_2 rand_2 \left(p^d_{lbest_i}(s) - x_i^d(s)\right) + C_3 rand_3 \left(p^d_{gbest}(s) - x_i^d(s)\right)$$

$$(7)\ x_i^d(s+1) = x_i^d(s) + v_i^d(s+1)\ ;\ d=1,...,m$$

that use some parameters: *w* is inertia weight, C_1, C_2 and C3 (see [5]) are acceleration coefficients; $rand_1$, $rand_2$, $rand_3$ are random numbers in the interval [0, 1].
A very efficient technique is to adapt the coefficients C_1, C_2 and C_3 and *w* along through the searching process (see [18]).

3. MODEL OF THE OPTIMIZED PROCESS

An optimal control problem has some constitutive elements that are recalled hereafter, trying to avoid certain mathematical details. A formulation of an OCP has as a central element the model of the dynamic system. This one is modeled by the general differential equation:

$$(8)\ \frac{dx}{dt} = f(x(t), y(t), u(t), t)$$

The variables used in this equation are as follows:

- *t:* the continuous time, $t \in \boldsymbol{R}$;
- $x(t)$: the vector of state variables;
- $y(t)$: the vector of algebraic variables (usually system output variables).
- $u(t)$: the vector of control variables.

The OCP is also defined by some *equality* and *inequality constraints* listed below:

- initial conditions: $x(t_0) = x^0$ (t_0: initial time);
- algebraic equality: $g\left(x(t), y(t), u(t), t\right) = 0$;
- path constraints:

$$h\left(x(t), \frac{dx(t)}{dt}, y(t), u(t), t\right) \le 0$$

- terminal equality constraints (t_f is the final time):

$$(9)\ \psi\left(x(t_f), \frac{dx(t_f)}{dt}, y(t_f), u(t_f), t_f\right) = 0$$

- bound constraints

$$(10)\ x^m \le x(t) \le x^M\ ;\ u^m \le u(t) \le u^M\ ;\ t_f^m \le t_f \le t_f^M\ ,$$

The superscripts *m* and *M* are associated with the minimum and maximum values, respectively.

The functioning of the system may be characterized through the value of a specific objective function $J\left(x(t_f), y(t_f), u(t_f), t_f\right)$. The OCP consists in finding the control variables *u* that met all the constraints and minimizes the objective function

$$(11)\ \min_{u(t), t_f} J\left(x(t_f), y(t_f), u(t_f), t_f\right)$$

4. IMPLEMENTATION USING HTPSO

The optimal solutions are searched in a specific space using a time horizon that is [0, t_f]. The time discretization yields a sequence of *n* time moments, usually equidistant (see [10]), which cover the time horizon:

$$(12)\ \bar{t} = (t_1, t_2, \cdots, t_n)^T;\ \text{with}\ t_n = t_f$$

The main unknown variables form together the control sequence $\bar{u}$ corresponding to these time moments, i.e. the so called *control profile*:

$$(13)\ \bar{u} = (u_1, u_2, \cdots, u_n);$$

Hence, the solution of an OCP $\bar{x}$ may be coded by

$$(14)\ \bar{x} = (\bar{u}, t_f)^T\ \text{ or }\ \bar{x} = \bar{u}^T$$

according as the final time is free (unknown) or fixed.

The HTPSO algorithm solving an OCP has some particularities in comparison with other cases when the problem has a different type. The first specificity is the fact that the objective function

evaluation is more complex. because A simulation of the dynamic system using the control profile set by the HTPSO algorithm is carried out. In the same time, the objective function value is computed.

Another specificity of the HTPSO algorithm is the integration of the *step control* technique. At two successive time moments, the control inputs values are, of course, randomly set. Hence, the difference between these values can be quite big. In most cases this fact doesn't fit with a certain smoothness of the control profile. That is why the following differential constraint must be implemented:

$$(15)\quad \left|\frac{du(t)}{dt}\right| \le \Delta s_{\max},$$

where $\Delta s_{\max}$ is the maximal accepted slope. Taking into account the codification of the solution $\bar{x}$, this means the limitation of the control input level at successive time moments, aiming to have a quasi-smoothness of the control profile. The adopted techniques are partially those used in the algorithms proposed in [3] and [9].

Finally, the algorithm returns the position of the particle that is the global best solution, namely the control profile for our OCP. We shall denote more explicitly this control profile by $u^{HTPSO}(.)$.

5. CASE STUDY

In order to illustrate how the HTPSO based algorithm and the control structure proposed in the next section work, we consider here after the case of a drive system with DC motor taken from the book [2]. The model of the DC motor is given by the equation

$$(16)\quad \ddot{\theta} = u(t),$$

where θ is the angular position and the control input $u(t)$ may be physically a current, voltage or motor torque. The problem is to find out the optimal command $u^*(t)$, $t\in[0, 2]$, which transfers the system from initial state (at t_0=0) $\theta_0 = 1$, $\dot{\theta}_0 = 1$ to final state (at t_f=2) $\theta_f = 0$, $\dot{\theta}_f = 0$, such that to minimize the *objective function J*:

$$(17)\quad J = \frac{1}{2}\int_0^{t_f} u^2(t)dt; \quad \min_u J$$

The state equations can be:

$$(18)\quad \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u(t) \end{cases}$$

with the initial state:

$$x(t_0) = [1 \quad 1]^T$$

and the final state:

$$x(t_f) = [0 \quad 0]^T.$$

This OCP is *bilocal* with fixed final time. The bilocal character can be treated by adding new penalizing terms in the objective function. In our case, the *extended objective function* may be

$$J_e = \left\{ c_1\left(x_1(t_f) - x_1^{t_f}\right)^2 + c_2\left(x_2(t_f) - x_2^{t_f}\right)^2 + \frac{1}{2}\int_0^{t_f} u^2(t)dt \right\},$$

and the optimality criterion becomes:

$$(19)\quad \min_u J_e$$

An important matter is to choose the constants c_1 and c_2 that multiply the penalizing terms (see [18]). For this case study, we chose c_1=c_2=50 (*static strategy*).

The HTPSO algorithm will work with the *extended objective function* J_e. At convergence, the algorithm returns the global best control profile $u^{HTPSO}(.)$ and the minimum value for the J_e. The theoretical optimal command and the state equations are:

$$(20)\quad u^*(t) = 3t - 3.5,$$

$$(21)\quad \begin{aligned} x_1^*(t) &= 0.5t^3 - 1.75t^2 + t + 1 \\ x_2^*(t) &= 1.5t^2 - 3.5t + 1 \end{aligned}$$

The HTPSO based algorithm has been executed many times, firstly for parameters calibration (see [18])

A typical evolution of the HTPSO algorithm leads to the results depicted in fig. 1 and fig. 2. Fig. 1 shows a very good evolution of the control profile in comparison with its theoretic behavior. The total number of objective function evaluations is N_{eval}=2420 for n=50.

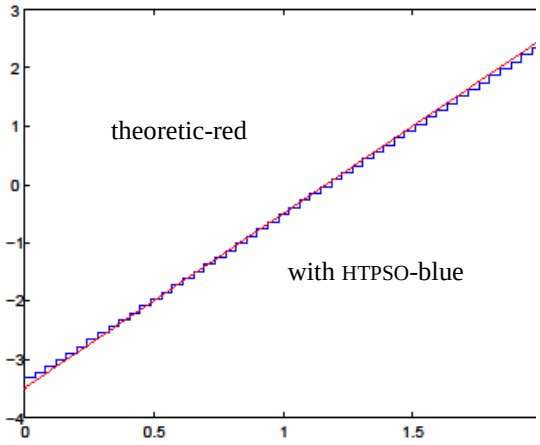

Fig. 1. Control profile evolution in open loop

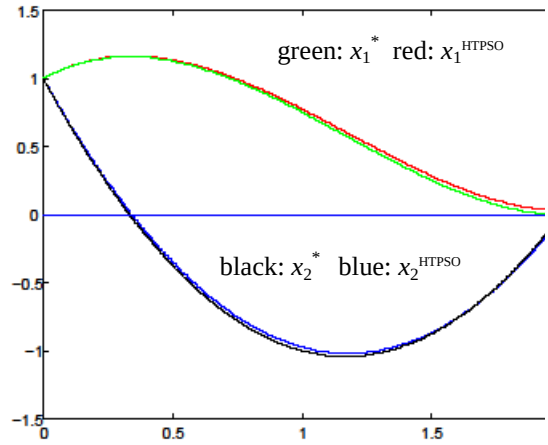

Fig. 2 State variables evolution in open loop

Let's notice that the algorithm has a very good convergence, because in only 121 steps the particle swarm converges to a very good solution.

The numerical complexity of the execution is very good as well. It is well-known that a good measure of the practical complexity of such an algorithm is the total number of objective function evaluations.

The fact that N_{eval}=2420 is the evidence of its remarkable complexity that is very small (compared with other metaheuristic).

The fig. 2 shows the state variables evolution on the interval $t \in [0,2]$ under the control profile $u^{HTPSO}(.)$, but also using the theoretic optimal control $u^*(.)$. The trajectories involved by the two evolutions are almost identical.

6. CONTROL STRUCTURE USING MODEL PREDICTIVE CONTROL

As we have already mentioned, an OCP solution obtained by the HTPSO algorithm is equivalent to an open loop control for the considered dynamic system. The main problem is to find a closed loop control structure that can use the HTPSO algorithm and take into consideration the real time values of the state or output vectors. Only in this way the perturbations acting on some variables can be rejected and the control objectives achieved.

Model Predictive Control (MPC), is a well known solution for this kind of problem, when a process model is available. In the sequel the aim of this work is to join the Model Predictive Control structure with the already developed HTPSO algorithm.

The control structure that uses MPC in the context described before is depicted in fig. 3. A *Process Model* (PM) of the Real Process tacking place in the dynamic environment is able to calculate in each moment the values of the some dependent variables as a result of prior variation of the decision variables. The environment changes are expressed by the values of the so-called state variables and controlled variables.

This control structure has two main characteristics:

- firstly, it introduces the closed loop structure that is crucial for a real-time application development;
- secondly, it can replace the optimization algorithm that is within the controller and minimizes the prediction error by the HTPSO algorithm. It can be shown that the HTPSO algorithm makes implicitly the minimization of the prediction error.

In the sequel, let's consider t_0=0 and the discrete moments $t_k=k\cdot T$ will be specified simply by k. So, the control horizon is the interval [0, H]. We introduce hereafter the following notations:

- [k, H] is the prediction horizon, with $k < H$, $k = 0, 1,\ldots, H$-1

- $U(k+i|k)$, $i=0,\ldots,H-k-1$ is the predicted value for $U(k+i)$ based on knowledge up to moment k.

- $X(k+i|k)$, $i=1,\ldots, H-k$ is the predicted value for $X(k+i)$ based on knowledge up to moment k;

Let's notice that

$$U(k+i|k) \neq U(k+i), \quad k>0\ , i=1,\ldots,H-k-1,$$

because $U(k+i|k)$ is a future value of the control input predicted at the present moment, whereas $U(k+i)$ is the future real value of the control input at the moment $k+i$ that obviously is not known at the present moment. The same thing can be asserted for the state variables.

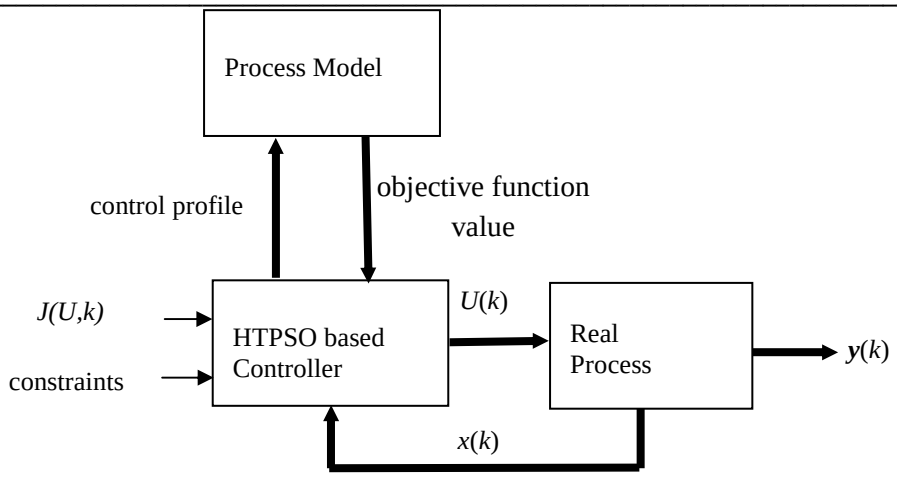

Fig. 3. Control Structure with PSO based Controller

Figure 3 suggests also how MPC involves the predictive control technique. This means that at present moment k when the state variable is $X(k)$, the performance index $J(U, k)$ for the interval $[k, H]$ is minimized subject to the constraints through an optimal control sequence:

$$< u^*(k|k),\ldots, u^*(H-1|k) >. \quad (31)$$

The first element $U(k)= u^*(k|k)$ of this sequence is really applied to the system. Then the left limit of the horizon is shifted one sample and the optimization is restarted for the interval $[k+1, H]$. The minimization is made within the Model Predictive Controller by the HTPSO algorithm, because it can be shown that the minimization of the objective function is equivalent to the minimization of the prediction error.

7. SIMULATION RESULTS

We shall prove the efficiency of this control structure through simulations using MATLAB system. A simulation covering the entire control horizon was repeated 30 times. The algorithm converged every time and supplied practically the same results for the closed loop behavior, in a small number of steps with a very small number of objective function evaluation.

First simulation series

The simulations use a Real Process Model that emulates the dynamic of the real process.

In the first phase of simulations, we consider the Real Process Model as being identical with the Process Model considered by the MPC structure. The typical evolution of the control structure using the MPC with HTPSO controller is described in fig. 4 (with blue line), versus the theoretical optimal control in open loop (OCP's solution with red line).

In fig. 5 the states' evolution of the closed loop with HTPSO controller is presented (red-x_1; blue-x_2) in comparison with theoretical optimal evolution in open loop (green: x_1; black: x_2)

The total number of objective function evaluations is 4740, which is satisfactory for 50 sampling periods. The analysis of the two figures 4 and 5 leads to the conclusion that the MPC with HTPSO controller is a realistic manner to implement a closed loop with very good results.

Second simulation series

In this phase, the Real Process Model is based on the Process Model but altering the two state variables - supposed to be accessible or that can be estimated - by an additional noise. This noise emulates the influences of the perturbations coming from the real process through the channel of state variables. In this work the noise is a random variable with uniform distribution in the interval $[-L_k^1, L_k^1]$ respectively $[-L_k^2, L_k^2]$, where:

$$L_k^1=0.05 \cdot |x_1(k)|; \qquad L_k^2=0.05 \cdot |x_2(k)|;$$

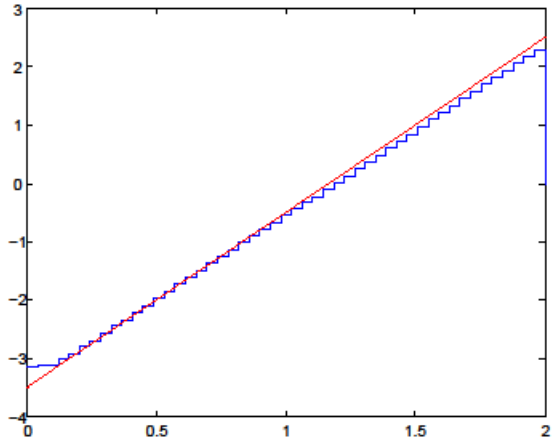

Fig. 4 Control input: theoretical and with HTPSO controller

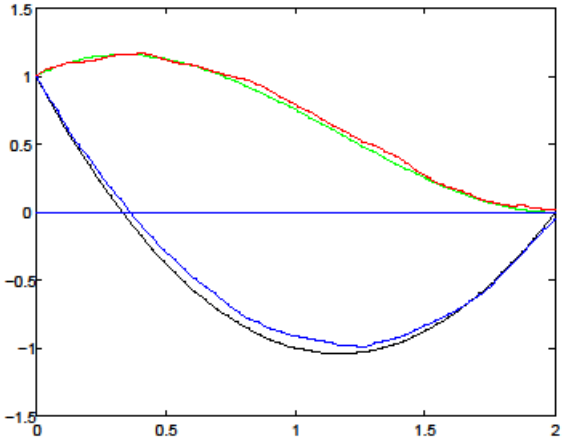

Fig. 5 State variables: theoretical and with HTPSO controller

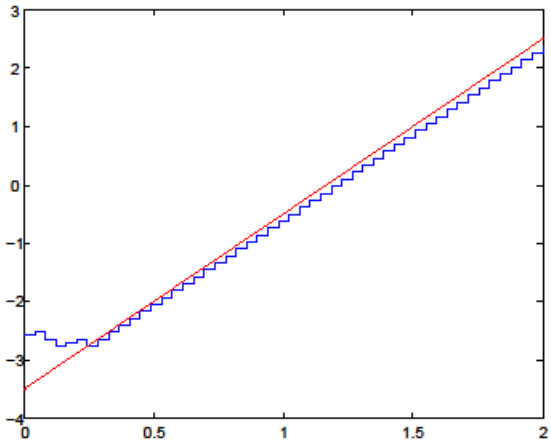

Fig. 6 Control input: theoretical and with a real process

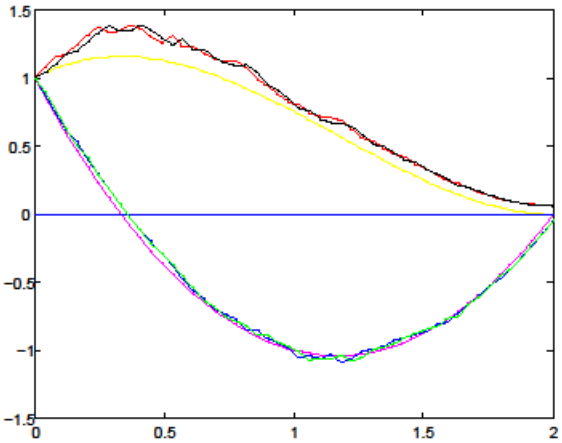

Fig. 7 State evolution: theoretical and with a real process

The fig. 6 and 7 present the results of the second simulation series. The figure 7 depicts the state evolution when the process is controlled in the following situations:

- in closed loop with a real process model (black-x_1; green- x_2);
- in closed loop with a simplified process model identical with that one used by the MPC (red-x_1; blue- x_2);
- in open loop -theoretical evolution- (yellow-x_1; magenta- x_2)

One can remark the similar evolution of the states in the case of closed loop based on HTPSO controller with or without additive noise. The total number of objective function evaluations is 5000, which is remarkable.

The analysis of the two figures 6 and 7 allow us to conclude that the closed loop structure with HTPSO controller and feedback from the real state variables is a realistic solution with very good results.

8. CONCLUSION

In a previous paper, a HTPSO based algorithm has been proposed in order to solve OCPs. The main problem of using the proposed algorithm is a systemic one: to use it in a closed loop control. Because a process model is generally speaking available, the MPC structure is the solution in conjunction with the idea of using the HTPSO based algorithm. This later one minimizes implicitly the prediction error.

Our case-study refers to a simple control problem whose theoretical solution can be easily deduced and used in a comparative analysis. The proposed

HTPSO algorithm has the ability to solve a large class of OCPs, whose process model is expressed by differential and algebraic equations and usual types of constraints.

The effectiveness and efficiency of the control structure in closed loop were proved by a set of simulations where the real process is replaced by a model. In the first series of simulations this model is even the process model used by the MPC structure. In this way, the simulations show the effect of loop closing. In the second series of simulations, the real process is implemented in a more realistic way by considering an additional noise to the state variables. The results prove that our approach is efficient and is good solution to our initial problem.

Because the HTPSO based algorithm has very good properties concerning the convergence and the computing complexity, the resulting MPC structure can solve efficiently other OCPs with larger complexity. In this way, the problem of implementing a real-time optima control structure has a realistic solution.

9. REFERENCES

Abraham, A.; L. Jain, R. Goldberg, Evolutionary Multiobjctive Optimization - *Theoretical Advances and Applications*, Springer ISBN 1-85233-787-7.

Belea, C.; *Teoria sistemelor* (System Theory), Editura Didactica si Pedagogica, Bucuresti 1985.

Beheshti, Z.; S. M. Shamsuddin, S. Hasan, "Memetic binary particle swarm optimization for discrete optimization problems", ELSEVIER, *Information Sciences* 299 (2015), p. 58-84.

Maurice, Clerc; *L'optimisation par essaims particulaires-versions paramétriques et adaptatives*, Hermes, Lavoisier, Paris 2005.

Chen, C-H.; J-C. Hwang and S-N. Yeh, "Personal Best Oriented Particle Swarm Optimizer", *Particle Swarm Optimization*, Edited by Aleksandar Lazinica, Published by In-Tech, ISBN 978-953-7619-48-0, 2009.

Clerc, M., J. Kennedy, "The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space", I*EEE Transactions on Evolutionary Computation*, vol. 6, p. 58-73, 2002.

Kennedy, J; Eberhard R., "Particle swarm optimization", in Proceedings of the *IEEE International Conference on Neural Networks*, vol. 4 (IEEE Press, Piscataway, NJ, USA, 1995), pp.942-1948.

Kennedy, J., Eberhart, R., Shi, Y., *Swarm Intelligence*, Morgan Kaufmann Academic Press, San Fransisco, 2001.

Kruse, R.; C. Borgelt, C. Braune, S. Mostaghim, M. Steinbrecher, *Computational Intelligence - A Methodological Introduction*, second edition, 2016, Springer.

Faber, R.; T. Jockenhövelb, G. Tsatsaronis, "Dynamic optimization with simulated annealing", *Computers and Chemical Engineering* 29 (2005) 273–290

Helwig, S.; J. Branke, S. Mostaghim, "Experimental analysis of bound handling techniques in particle swarm optimization", *IEEE Trans. Evol. Comput*. 17(2), 259-271, 2013.

Mînzu, V.; M. Barbu, *et al.*, "A Binary Hybrid Topology Particle Swarm Optimization Algorithm for Sewer Network Discharge", *Proceedings of the 19th ICSTCC*, Cheile Gradistei, Romania, October 14-16, 2015, ISBN: 978-1-4799-8480-0 ©2015 IEEE, pp 627-634.

Onwubolu, G.; B.V. Babu, *New Optimization Techniques in Engineering*, Springer, ISSN 1434-9922.

Siarry Patrick-Editor, *METAHEURISTICS* , Springer, ISBN 978-3-319-45401-6, 2014, ISBN 978-3-319-45403-0 (eBook), 2016.

Talbi, E. G.; *METAHEURISTICS-From Design to Implementation*, ISBN 978-0-470-27858-1, WILEY, 2009.

Valadi; J. and P. Siarry -editors, *Applications of Metaheuristics in Process Engineering*, ISBN 978-3-319-06507-6, Springer, 2014.

Maurice, C.; *L'optimisation par essaims particulaires-versions paramétriques et adaptatives*, Hermes, Lavoisier, Paris 2005;

Mînzu, V.; Optimal Control Using Particle Swarm Optimization, International , The 5th IEEEE International Symposium on Electrical and Electronics Engineering, 20-22 October, Galati, Romania