Particle Swarm Optimization for Nonlinear Model Predictive Control

Julian Mercieca and Simon G. Fabri
Department of Systems and Control Engineering
University of Malta
Msida, Malta
Email: julianmercieca@gmail.com, simon.fabri@um.edu.mt

***Abstract*—The paper proposes two Nonlinear Model Predictive Control schemes that uncover a synergistic relationship between on-line receding horizon style computation and Particle Swarm Optimization, thus benefiting from both the performance advantages of on-line computation and the desirable properties of Particle Swarm Optimization. After developing these techniques for the unconstrained nonlinear optimal control problem, the entire design methodology is illustrated by a simulated inverted pendulum on a cart, and compared with a particular numerical linearization technique exploiting conventional convex optimization methods. This is then extended to input constrained nonlinear systems, offering a promising new paradigm for nonlinear optimal control design.**

***Index Terms*—particle swarm optimization; model predictive control; optimal control**

I. Introduction

Nonlinear Model Predictive Control (NMPC) is an attractive control scheme for manipulating the behaviour of complex systems [1], exhibiting excellent dynamic performance in both industrial applications and theoretical studies [2]–[4]. However its application to nonlinear control is complicated, largely due to the optimization method that has come to be used in these controllers. A fundamental difficulty of the NMPC approach is the requirement to solve nonconvex constrained optimization problems. Most existing works are based on nonlinear programming methods [5] which only yield local optimum values, with the latter depending on the selection of the starting point. For this purpose, in [6] a particular numerical linearization technique has been developed to obtain a convex constrained optimization problem, albeit at the cost of performance deterioration. Other attempts to solve the nonconvex optimization problems exploit Genetic Algorithm (GA) optimizers [7]. However these face many challenges, including enormous computational effort due to its natural genetic operations [8], [9]. Although this may be reduced by using a real-value representation in the GA [8], [10], some deficiencies in GA performance have been highlighted in recent research. Applications governed by highly epistatic objective functions [11], [12] reveal shortfalls in performance, which is further worsened by the GA's premature convergence [11].

This paper presents two novel NMPC controllers based on a very powerful optimizer: Particle Swarm Optimization (PSO). First developed by Kennedy and Eberhart in 1995 [13], this modern metaheuristic algorithm has been found to be robust in solving continuous nonlinear optimization problems [10], [12]–[14] and capable of generating high quality solutions with more stable convergence characteristics and shorter calculation times than other stochastic methods [10], [12], [15]. One of the novel controllers presented in this paper is based on a numerical linearization technique first proposed by Alaniz in [6], which is based on conventional convex optimization methods. By contrast, the proposed controller exploits PSO techniques for optimization. The second novel controller proposed in this paper does away with any form of numerical linearization to achieve optimization of the cost function. Both controllers are simulated for an inverted pendulum on cart problem and compared with the NMPC controller in [6].

The rest of the paper is organized as follows. Section II is a brief explanation of the implemented PSO algorithm, while Section III outlines the design of the three NMPC controllers evaluated in this paper. Section IV then presents the simulation setup, results and analysis, followed by a brief conclusion in Section V.

II. Particle Swarm Optimization

The particle swarm optimization (PSO) algorithm [13] is a population-based search algorithm inspired by the social behaviour of birds within a flock. The very simple behaviour followed by individuals emulates their own successes and the success of neighbouring individuals. The emergent collective behaviour is that of discovering optimal regions of a high dimensional search space.

In a PSO algorithm, each particle representing a potential solution is maintained within a swarm. In simple terms, the particles are therefore "flown" through a multidimensional search space where the position of each particle is adjusted according to the experience of itself and its neighbours. Let $\mathbf{x}_i(t)$ denote the position of particle i in the search space at time step t, which denotes discrete time steps unless otherwise stated. The position of the particle is changed by adding a velocity vector, $\mathbf{v}_i(t)$, to the current position *i.e.*

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1) \tag{1}$$

with $\mathbf{x}_i(0) \sim U(\mathbf{x}_{min}, \mathbf{x}_{max})$, where $U(\mathbf{x}_{min}, \mathbf{x}_{max})$ denotes the continuous uniform probability distribution within the real-valued space $(\mathbf{x}_{min}, \mathbf{x}_{max})$. The optimization process is driven

 ISBN: 978-1-61208-172-4

by the velocity vector, reflecting both the experiential knowledge of the particle (*cognitive component*) and socially exchanged information from the particle's neighbourhood (*social component*). In this paper we implement a particular PSO algorithm know as global best PSO, which exhibits very fast convergence rates [16] much needed for our predictive control application. For the global best PSO, or *gbest* PSO, the neighbourhood for each particle is the entire swarm, thus employing the social network of the star topology type. In this situation, the social information is the best position found by the swarm, referred to as $\hat{\mathbf{y}}(t)$.

For *gbest* PSO, the velocity of particle i is calculated as

$$\begin{aligned} v_{ij}(t+1) = \quad & v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + \\ & c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)] \end{aligned} \tag{2}$$

where $x_{ij}(t)$, $y_{ij}(t)$ and $v_{ij}(t)$ are the position, personal best position and velocity of particle i in dimension $j = 1, \ldots, n_x$ at time step t respectively, $\hat{y}_j(t)$ is the global best position in dimension j, c_1 and c_2 are positive acceleration constants used to scale the contribution of the cognitive and social components respectively, and $r_{1j}(t), r_{2j}(t) \sim U(0,1)$ are random values in the range $[0,1]$, sampled from a continuous uniform distribution. These random values introduce a random element to the algorithm. Algorithm 1 summarizes the *gbest* PSO, where $\mathbf{y}_i$ denotes the personal best position associated with particle i and $\hat{\mathbf{y}}$ denotes the global best position.

Algorithm 1 *gbest* PSO

Create and initialize an n_x-dimensional swarm
repeat
  for *each particle* $i = 1, \ldots, n_s$ **do**
    //set the personal best position
    if $f(\mathbf{x}_i) < f(\mathbf{y}_i)$ **then**
      $\mathbf{y}_i = \mathbf{x}_i$;
    end
    //set the global best position
    if $f(\mathbf{y}_i) < f(\hat{\mathbf{y}})$ **then**
      $\hat{\mathbf{y}} = \mathbf{y}_i$;
    end
  end
  for *each particle* $i = 1, \ldots, n_s$ **do**
    update the velocity using equation (2);
    update the position using equation (1);
  end
until *stopping condition is true*;

III. NONLINEAR MODEL PREDICTIVE CONTROL

A nonlinear dynamic system may be represented by a set of nonlinear differential equations [17], which may be discretized for computational purposes using Euler's method, where T_s is the sampling period and k is the sample index in discrete-time, as follows:

$$\mathbf{x}(k+1) = \mathbf{x}(k) + T_s f(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k), k) \tag{3}$$

$$\mathbf{y}(k) = g(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k), k) \tag{4}$$

Arguments of the nonlinear function f include a state vector $\mathbf{x}(k)$, a control input $\mathbf{u}(k)$, and a disturbance input $\mathbf{v}(k)$. The set of physical quantities that can be measured from the system constitute the output, $\mathbf{y}(k)$, which is also a nonlinear function g of the same arguments. More accurate discretization approximations, such as the Runge-Kutta methods, can be used if the system dynamics are highly nonlinear or the desired time step is large.

The Model Predictive Control (MPC) design methodology is characterized by three main features: an explicit model of the plant, computation of control signals by optimizing predicted plant behaviour, and a receding horizon [18]. An internal model is used to predict how the plant will react, starting at the current time k, over a discretized prediction interval. The objective is to select the control history that results in the best predicted behaviour with respect to a reference trajectory and optimization parameters.

The cost function used in this paper is given by equation (5) which has a quadratic structure comprising two terms. The first term, weighted by a symmetric weighting matrix $\mathbf{Q}(k)$, penalizes the deviations from a reference trajectory that occur throughout the prediction interval. The second term, weighted by a symmetric weighting matrix $\mathbf{R}(k)$, penalizes the magnitude of each control value in the control history. We will now describe the three nonlinear model predictive controllers considered in this paper, two of which represent the novel contributions of this work.

$$J = \sum_{i=0}^{l-1} ||(\mathbf{y}(k+i) - \tilde{\mathbf{y}}(k+i))||^2_{\mathbf{Q}(k+i)} + \sum_{i=0}^{m-1} ||\mathbf{u}(k+i)||^2_{\mathbf{R}(k+i)} \tag{5}$$

A. A numerical linearization method

This method, proposed by Alaniz in [6], centres around a particular numerical linearization technique for generating the predicted output trajectory $\mathbf{y}$. A nominal control history $\bar{\mathbf{u}}$ is first chosen, then the corresponding nominal output trajectory $\bar{\mathbf{y}}$ is computed through numerical integration. Typically $\bar{\mathbf{u}}$ is the previous optimal solution, but it can be set equal to zero if none exist. The predicted output is then based on linearizing the control perturbation $\Delta\mathbf{u}$ about the nominal trajectory as follows:

$$\begin{aligned} \mathbf{y}(k) &= \bar{\mathbf{y}}(k) + \alpha_0 \Delta\mathbf{u}(k) \\ \mathbf{y}(k+1) &= \bar{\mathbf{y}}(k+1) + \alpha_1 \Delta\mathbf{u}(k) + \beta_0 \Delta\mathbf{u}(k+1) \\ \mathbf{y}(k+2) &= \bar{\mathbf{y}}(k+2) + \alpha_2 \Delta\mathbf{u}(k) + \beta_1 \Delta\mathbf{u}(k+1) + \gamma_0 \Delta\mathbf{u}(k+2) \\ &\vdots \end{aligned} \tag{6}$$

The coefficients α_i, β_i, γ_i, ... are produced by computing a perturbed trajectory for each $\Delta\mathbf{u}(k+i)$ and finding the subsequent deviation from the nominal trajectory. Perturbed trajectories are the result of adding a pulse of magnitude one to the nominal control history at time $(k+i)$. Each trajectory is formed by propagating the present state $\mathbf{x}(k)$ over a fixed interval of time while applying an associated control history. The prediction interval and control history are divided into l and m discrete steps, respectively, of length T_s, where T_s is the

 ISBN: 978-1-61208-172-4

time step, and $m \leq l$. After the control history has ended, the control is held constant for the final $(l-m)$ time steps.

The MPC problem is to solve for the optimal control perturbation $\Delta\mathbf{u}^*$ by minimizing a cost function with respect to a reference trajectory and optimization parameters. The optimal control history is then the sum of the nominal control history and the optimal control perturbation [6]. By rearranging and simplifying the form of equation (5), a set of matrices is obtained which leads to the unconstrained and constrained optimization problems. For the unconstrained case, Alaniz [6] presents a solution by using an equivalent least squares technique, while for the constrained case, the problem is reinterpreted so as to obtain the standard form handled by quadratic programming solvers.

Once the optimal control history is chosen, the first N time steps of the solution are applied to the plant. The cycle of forming predicted behaviours and solving for the optimal control perturbations is then repeated using the most recent feedback from the plant. The interested reader is referred to [6] for further detail about this technique.

B. A novel numerical linearization technique using PSO

A novel application of PSO proposed here exploits the aforementioned numerical linearization technique used in conjunction with the PSO algorithm, where the convex least squares or quadratic programming optimization methods are now replaced by the global best PSO algorithm. The evaluation function is the cost given by equation (5), so that PSO searches for the optimal perturbed control history of equation (6), denoted by $\Delta\mathbf{u}(k)^*$, in order to obtain the optimal control history $\mathbf{u}(k)^*$ that minimizes J. For this purpose, we require an m-dimensional PSO, with each particle's position defined by $\mathbf{K}$, an m-dimension column vector equal to $\Delta\mathbf{U}(k)^*$, which is a column vector having $\Delta\mathbf{u}(k+i)^*$ as its elements.

C. A novel PSO-based nonlinear MPC strategy

The second novel controller makes use of the PSO search algorithm for obtaining the optimal control history that minimizes directly the cost function J given by equation (5) without resorting to numerical linearization as represented by equation (6). In this manner we simply use equation (5) as the evaluation function to be minimized using global best PSO, thereby avoiding any linearization technique or mathematical result for minimization, albeit at an increased computational complexity. Each particle's position in the swarm represents the m-dimension column vector defining the optimal control history, $\mathbf{U}(k)^*$.

As we shall see, this remarkably simple method produces the best results for the controllers studied in this paper in terms of the performance index obtained. The block diagram in Figure 1 illustrates the structure of the proposed predictive control loop. A particle swarm optimizer uses the reference input and predicted output trajectories to minimize the quadratic cost function given by equation (5) and compute the optimal control history which is then applied to the plant. The proposed controller is further enhanced by actively

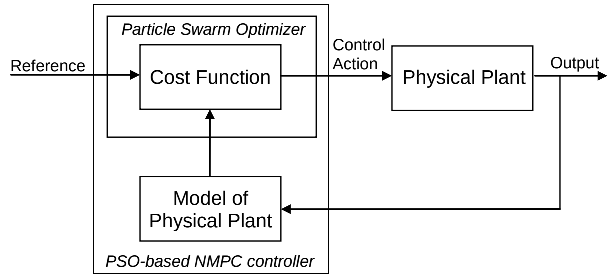


Fig. 1. PSO-based nonlinear MPC loop

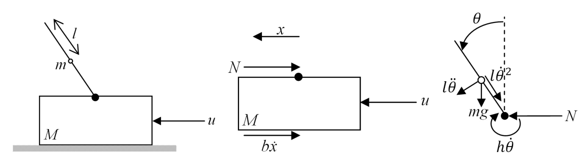
correcting the weighting matrix $\mathbf{R}$ in an adaptive manner, so that the chattering effect of the control input observed about the equilibrium point is reduced.

IV. Performance Evaluation: Inverted Pendulum on Cart

The performance of the proposed controllers is evaluated by analyzing the results from simulation experiments. The plant chosen for simulation is an inverted pendulum on a cart and two types of controllers are generated for the three methods presented in this paper; an unconstrained and constrained version. The latter problem shall only consider single constraints and therefore no penalty functions are required. The pendulum is initially at the stable equilibrium point and the purpose of each controller is to invert the pendulum. Since the dynamics at the stable and unstable equilibrium points are very different, this is a good problem to demonstrate the effectiveness of our nonlinear MPC controllers.

A. Plant Model

The nonlinear model of the plant is derived by applying Newton's Laws of Motion to the free body diagrams in Figure 2. The resulting equations of motion are given by equations (7) and (8). A complete derivation is given in [6].



(a) (b) (c)

Fig. 2. (a) Inverted Pendulum on a Cart; (b) Free body diagram 1; (c) Free body diagram 2.

$$\ddot{x} = \frac{1}{M+m}\left[u - b\dot{x} - ml\ddot{\theta}\cos(\theta) + ml\dot{\theta}^2\sin(\theta)\right] \quad (7)$$

$$\ddot{\theta} = \frac{3}{4ml^2}\left[mgl\sin(\theta) - ml\ddot{x}\cos(\theta) - h\dot{\theta}\right] \quad (8)$$

M is the mass of the block that slides along a surface, m is the uniformly distributed mass of an ideal pendulum, $2l$ is the length of the ideal pendulum, b is the surface friction damping constant, h is the rotational friction damping coefficient, u is the force applied to the block, θ is the clockwise angle between the normal and the pendulum (as shown in Figure 2(c)), and x is the cart's horizontal displacement from its

 ISBN: 978-1-61208-172-4

equilibrium position. To allow the model to be numerically integrated, equations (7) and (8) are expressed in terms of the state variables x, $\dot{x}$, θ, and $\dot{\theta}$. The second-order differential equations are then discretized using the fourth-order Runge-Kutta method [6].

B. Controller Layout

The simulation experiments were run on the Simulink software package [19]. The layout shown in Figure 3 is the simulated realization of the control loop given in Figure 1. It makes use of S-Functions that implement constrained or unconstrained versions of the PSO-based NMPC controller.

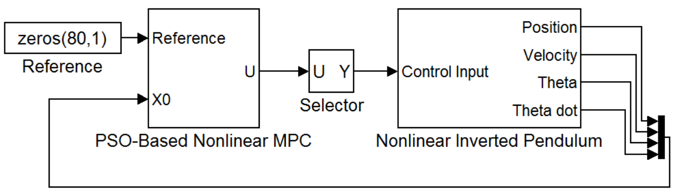


Fig. 3. Nonlinear MPC Simulink layout

C. Controller Parameters

The MPC controller rate is $\frac{1}{NT_s}$, where N is the number of controls in the control history that are applied to the plant. $N = 1$ is used in the controller since this is the typical value selected [18]. The computational load of MPC can be reduced if N is increased, but a disadvantage to having $N > 1$ is that some of the controls applied to the plant are based on old feedback. The fourth-order Runge-Kutta method is tested using different values for T_s, and it is established that the response with $T_s = 0.1s$ is almost indistinguishable from the actual response, thus using this value for the controller.

Since this controller is very computationally intensive, it is not feasible to have a long prediction length or control history. A value of $l = m = 20$ is chosen as a balance between performance and computation time. This results in a controller capable of predicting for 2 seconds.

The two novel PSO-based controllers use the following PSO parameters, which were derived empirically through successive simulations:

- Each particle consists of 20 members, corresponding to the 20 elements that make up the optimal control perturbation history column vector, $\Delta\mathbf{U}(k)^*$, for the PSO-based numerical linearization method, or the optimal control history column vector, $\mathbf{U}(k)^*$, for the PSO-based NMPC controller.
- Swarm size, n_s = 30.
- Inertia weight starting with $w(0) = 0.9$ and linearly decreasing to $w(n_t) = 0.4$.
- Velocity clamped to within half the particles' positional constraints.
- Search space is limited to real-valued variables between $-300N$ ($x_{min,j}$) and $300N$ ($x_{max,j}$) for the unconstrained case.
- Acceleration coefficients $c_1 = 2$ and $c_2 = 2$.
- Number of iterations = 30.

Both weighting matrices $\mathbf{Q}$ and $\mathbf{R}$ given in equation (5) are set equal to the values shown in equation (9). Deviations are measured from the reference trajectory which is set equal to a zero column vector. There is a zero for each state variable at each time step in the prediction interval. This reference remains constant for the duration of the simulation.

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; R = 1 \tag{9}$$

D. Simulation Results

The response of the unconstrained controller, using the three control schemes described in this paper, is shown in Figure 4 with the pendulum initially at the stable equilibrium point (180 degrees), hanging straight down. The running performance index plot describes the minimized value of the cost function for each time step. The cart's position moves back and forth so that the pendulum gains momentum. This continues until there is enough momentum to swing up and invert the pendulum in the 0 degree position. Table I shows the superior performance obtained for the proposed PSO-based NMPC controller for the unconstrained case. The performance index values quoted here are obtained using equation (5), this time using the actual output trajectory and the applied control inputs, summed over a sufficiently large amount of time (10s). This reveals the true performance index for the whole control action. When PSO is used in conjuction with the numerical linearization technique, no significant advantage is obtained over the least squares method (an improvement in J of only 1.46%), as expected for the convex optimization problem being solved. On the other hand, the second proposed nonlinear PSO controller gives an improvement in J of 8.04% over the numerical linearization (least squares) counterpart.

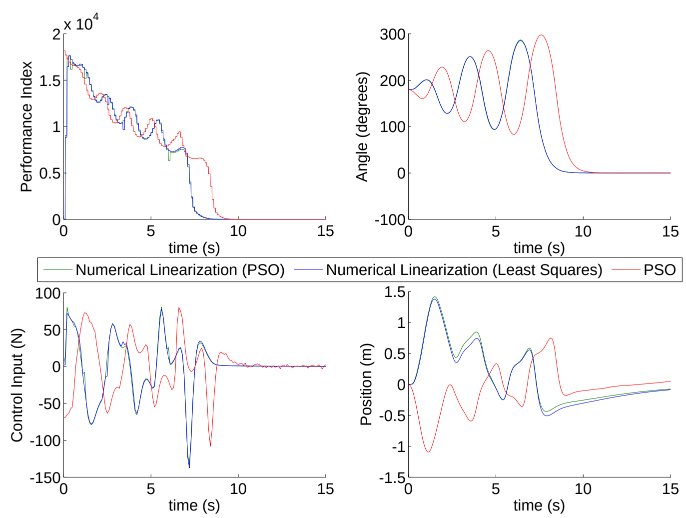


Fig. 4. Unconstrained nonlinear MPC: A comparison

Figure 5 plots the response of the constrained controller when a single constraint, restricting the control input of the

TABLE I
UNCONSTRAINED NMPC: PERFORMANCE INDEX VALUES

Method	Numerical Linearization (Least Squares) [6]	Numerical Linearization (PSO)	PSO
Performance Index J ($\times 10^6$)	1.4538	1.4326	**1.3369**

cart to be within $-45N$ and $45N$, is active. In Figure 5, the final angular deflection is either $0°$ or $360°$. Note that both these angles correspond to the same inverted position of the pendulum. For the novel PSO-based NMPC controller, the cart is noted to move a much smaller distance to achieve swing-up. In real-world terms, this translates to a more efficient process, with less work being done by the cart to achieve swing-up and equilibrium. This is further evidenced by Table II, which indicates that the novel PSO-based nonlinear MPC controller has the edge over the numerical linearization technique which uses quadratic programming, a method known to have the problem of getting stuck in local minima [20]. We record a 14.78% improvement in J, accompanied by a very low standard deviation when the experiment is repeated over 10 trials indicating PSO's repeatable nature despite being a metaheuristic optimization method. Note that for the constrained case, using the numerical linearization technique in conjunction with PSO is computationally inefficient since every particle must be checked for its corresponding optimal control history, doubling the workload of its unconstrained counterpart, rendering it practically useless to investigate for this purpose. The advantage of the novel PSO-based NMPC

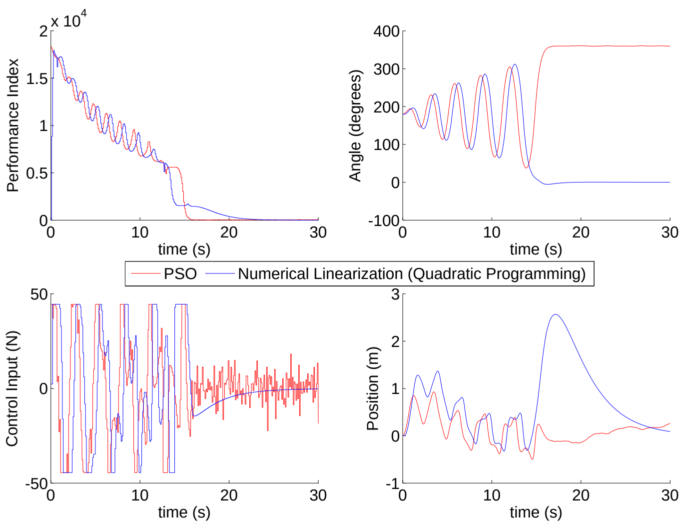


Fig. 5. Constrained nonlinear MPC: Restricting control input to within $-45N$ and $45N$ (10 independent trials)

TABLE II
CONSTRAINED NONLINEAR MPC: PERFORMANCE INDEX VALUES

Method	Numerical Linearization (Quadratic Programming) [6]	PSO (mean J)	PSO (standard deviation) ($\times 10^6$)
Performance Index J ($\times 10^6$)	2.8534	**2.4316**	**0.02396**

controller is even more evident in Figure 6, where the proposed active correction for the chattering effect of the control input is implemented for the same constrained NMPC problem by changing $\mathbf{R}$ dynamically. The control input is being more heavily penalized when the angle approaches the equilibrium point by increasing R from 1 to 30. In other words, we are telling the system that in the close neighbourhood of the equilibrium point, minimal control effort is required, mitigating the effect of metaheuristic stochasticity. This reduces the performance index even further, giving an improvement in J of 16.65% (see Table III), making the process even more efficient. The system's robustness to model uncertainty is best

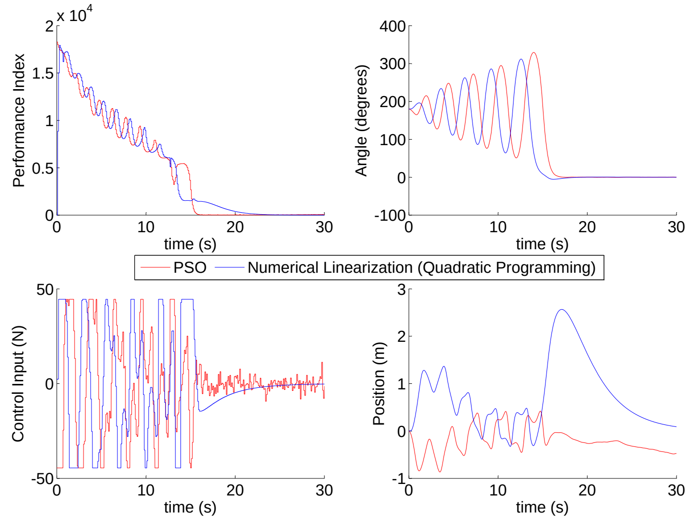


Fig. 6. Actively controlling control input weight R for reduced chattering.

TABLE III
PERFORMANCE INDEX VALUE COMPARISON FOR ACTIVE R CORRECTION

Method	Numerical Linearization (Quadratic Programming) [6]	PSO
Performance Index J ($\times 10^6$)	2.8534	**2.3810**

illustrated by the simulation results of Figure 7. This is tested by randomly increasing or decreasing each of the plant model's parameters by 5% (all parameters are changed for every trial), so despite the fact that the constrained NMPC controller is using a severely inaccurate model for its predictions, and we are not actively controlling the control input weight R (to consider the worst case), excellent performance is noted, and the pendulum swings up normally, except for a larger distance now required. Table IV shows the corresponding changes implemented in the model's parameters for the simulation results of Figure 7. The corresponding performance index values are given in Table V, where although both controllers manage swing-up and equilibrium similarly as for the results shown in Figure 5, the novel PSO-based NMPC controller exhibits an improvement in J of 12.07%. Repeatability is tested by performing several trials with different constraints, as shown in Table VI. The novel PSO nonlinear controller shows consistently better performance, with a mean improvement in J of 9.17%.

ISBN: 978-1-61208-172-4

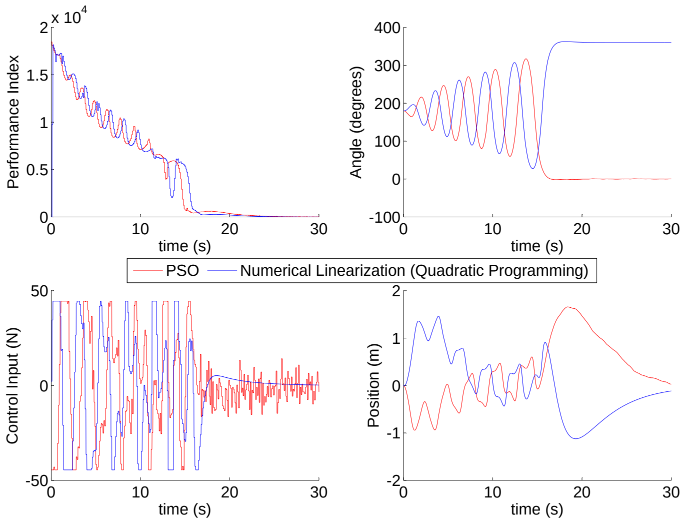


Fig. 7. Robustness Test: Model's parameters are significantly different from the actual plant parameters (constrained NMPC problem results shown).

TABLE IV
ACTUAL PLANT AND MODEL PARAMETERS (FOR A PARTICULAR TRIAL)

Parameter	Units	Actual Plant	Model
M	Kg	14.6	15.33
m	Kg	7.3	6.935
$2l$	m	2.4	2.52
b	Kg/s	14.6	15.33
h	Kgm^2/s	0.0136	0.0129

TABLE V
PERFORMANCE INDEX VALUES OBTAINED FOR THE ROBUSTNESS TEST

Method	Numerical Linearization (Quadratic Programming) [6]	PSO
Performance Index J ($\times 10^6$)	2.7369	**2.4065**

TABLE VI
SIMULATION RESULTS FOR DIFFERENT CONSTRAINTS (10 INDEPENDENT TRIALS WITH CONSTANT R)

Constraint	Numerical Linearization (Quadratic Programming) [6]	PSO
$-30N \leq U \leq 30N$	2.7182	**2.4026**
$-35N \leq U \leq 35N$	2.5374	**2.3947**
$-40N \leq U \leq 40N$	2.4590	**2.3880**
$-45N \leq U \leq 45N$	2.8534	**2.3810**

V. CONCLUSION AND FUTURE WORK

In this paper two novel controllers were proposed for the receding horizon strategy, both exploiting the desirable optimization properties of PSO. One makes use of the numerical linearization technique where instead of convex optimization methods, we employed a PSO strategy. The latter yielded a minor improvement in performance index over its convex optimization counterpart for a simulated inverted pendulum on cart problem. However the second novel PSO-based controller proved superior to both, approaching up to 16% less performance cost at best. In addition, we proposed a further enhancement for this novel controller by actively controlling the control input weight **R** to reduce the chattering effect of the control input observed for the nonlinear model predictive controller. Having shown that this framework extends to input constrained systems, we have provided the foundation to include other advances in control theory as they become available. Further work may include investigation of the use of PSO to obtain the much needed connection between the selection of weighting matrices **Q** and **R**, and performance specifications, possibly through some time-domain performance criterion. A similar investigation may be carried out for other control schemes, including linear quadratic optimal control strategies.

REFERENCES

[1] C. V. R. D. Q. Mayne, J. B. Rawlings and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789 – 814, 2000.
[2] E. F. Camacho and C. A. Bordons, *Model Predictive Control in the Process Industry*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1997.
[3] D. W. Clarke, *Advances in Model-Based Predictive Control*. Oxford University Press, 1994.
[4] K. R. Muske and J. B. Rawlings, "Model predictive control with linear models," *AIChE Journal*, vol. 39, no. 2, pp. 262–287, 1993.
[5] S. Shin and S. Park, "Ga-based predictive control for nonlinear processes," *Electronics Letters*, vol. 34, no. 20, pp. 1980 –1981, oct 1998.
[6] A. Alaniz, "Model predictive control with application to real-time hardware and a guided parafoil," Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, USA, 2004.
[7] X. Blasco, M. Martinez, J. Senent, and J. Sanchis, "Generalized predictive control using genetic algorithms (gagpc). an application to control of a non-linear process with model uncertainty," in *Methodology and Tools in Knowledge-Based Systems*, ser. Lecture Notes in Computer Science.
[8] T. Kawabe and T. Tagami, "A real coded genetic algorithm for matrix inequality design approach of robust pid controller with two degrees of freedom," in *Intelligent Control, 1997. Proceedings of the 1997 IEEE International Symposium on*, jul 1997, pp. 119 –124.
[9] R. Krohling, H. Jaschek, and J. Rey, "Designing pi/pid controllers for a motion control system based on genetic algorithms," in *Intelligent Control, 1997. Proceedings of the 1997 IEEE International Symposium on*, jul 1997, pp. 125 –130.
[10] P. Angeline, "Using selection to improve particle swarm optimization," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, may 1998, pp. 84 –89.
[11] D. B. Fogel, *Evolutionary computation: toward a new philosophy of machine intelligence*. Piscataway, NJ, USA: IEEE Press, 1995.
[12] R. C. Eberhart and Y. Shi, "Comparison between genetic algorithms and particle swarm optimization," in *Proceedings of the 7th International Conference on Evolutionary Programming VII*, ser. EP '98. London, UK: Springer-Verlag, 1998, pp. 611–616.
[13] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, nov/dec 1995, pp. 1942 –1948 vol.4.
[14] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, may 1998, pp. 69 –73.
[15] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," *Power Systems, IEEE Transactions on*, vol. 15, no. 4, pp. 1232 –1239, nov 2000.
[16] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, 2nd ed. Wiley Publishing, 2007.
[17] J. J. E. Slotine and W. Li, *Applied Nonlinear Control*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1991.
[18] J. Maciejowski, *Predictive control: with constraints*. Prentice-Hall, Harlow, UK, 2002.
[19] The MathWorks, Inc. (2011) Simulink - simulation and model-based design. [Online]. Available: http://www.mathworks.com/products/simulink/
[20] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.