DE GRUYTER OPEN

Journal of ELECTRICAL ENGINEERING, VOL 68 (2017), NO4, 290–298

Reducing usage of the computational resources by event driven approach to model predictive control

Stefan Misik, [*] **Zdenek Bradac,** [*] **Arben Cela** [**]

This paper deals with a real-time and optimal control of dynamic systems while also considers the constraints which these systems might be subject to. Main objective of this work is to propose a simple modification of the existing Model Predictive Control approach to better suit needs of computational resource-constrained real-time systems. An example using model of a mechanical system is presented and the performance of the proposed method is evaluated in a simulated environment.

K e y w o r d s: model-based control, sparse control, model predictive control, event-triggered control

1 Introduction

Automatic control plays important role in both industrial and commercial engineering applications. Most of these applications, nowadays, use digital computers in forms ranging from low-power single-chip controllers up to powerful personal computers with several processing cores. Limit in computing power of computers used in engineering applications, such as dynamic systems control, induces constraints on design of the control algorithms these computers can run. The constraints caused by limitations in real computer system is especially necessary to take into account when designing real-time application, like automatic control of mechanical systems. In real-time computer systems each task must respond to events within specified deadlines to avoid failure.

In traditional control design, which are usually designed using linear input/output models, on hard real-time computer system each task implementing a digital controller has its deadlines defined by period of execution and measured or calculated worst-case execution time (WCET). In simple designs these limitations of real computer system either do not constitute significant constraints on the design of the digital controller, or it is possible to incorporate portion of these limitations (*eg* communication and computational delay) into model of the controlled dynamic system [1].

In some cases a requirement may arise to use more advanced optimal control algorithms, such as Model Predictive Control, to achieve optimality and better control performance. Different methods used to design such controller can be found in the automatic control textbooks [1, 2]. In this case limitations of the real computer system may pose significant constraints on designed controller. One of the most critical of them, especially when real-time systems are of concern, may be the computational complexity. This issue can be addressed by different approaches, one of such method is introduced in [3] and [4]. In [4] is presented optimal control and scheduling designed on constrained computer system using both on-line and off-line technique. In this work author advocates usage of the H_2 norm and *periodic control theory* to design off-line periodic controller for resource constrained computer system. Also an on-line approach for improving performance of the off-line control and resource schedule, called *Optimal Pointer Placement*, is presented in this work. Interesting approaches can also be found in the literature dealing with the networked control systems (NCS), two main approaches oriented towards achieving resource utilization reduction (to save computation and communication resources) can be distinguished, namely, Event-Triggered Control (ETC) and Self-Triggered Control (STC) [5, 6]. The control law in ETC and STC consists of a feedback controller that computes the control signals, and a triggering mechanism that determines when the control signals have to be updated. The difference between ETC and STC is that in the former the control update is triggered by specific condition which is continuously being checked and when it becomes true, the control signals are recalculated, while in the latter the next update time is determined at current update time. The paper [5] introduces a general framework for the self-triggered MPC strategy applying to discrete-time nonlinear systems subject to state and input constraints and possibly a non-quadratic cost function. The framework proposed in [5] also provides a priori closed-loop performance guarantees in terms of original cost function apart form asymptotic stability and constraint satisfaction. If the messages in the communication network under consideration provide high space for useful data (payload), another mechanism to reduce utilization of communication resources (without resorting to sporadically chang-

* Department of Control and Instrumentation, Faculty of Electrical Engineering and Communication, Brno University of Technology, Brno, Czech Republic, stefan.misik@phd.feec.vutbr.cz, bradac@feec.vutbr.cz, ** Department of Computer Science and Telecommunication, Université Paris-Est, ESIEE Paris, Noisy le Grand, France, arben.cela@esiee.fr

DOI: 10.1515/jee-2017–0041, Print (till 2015) ISSN 1335-3632, On-line ISSN 1339-309X © 2017FEI STU

ing control signals) is at hand – *ie packet-based* predictive control. This approach exploits predictive nature of some control methods (*eg* MPC) to include predicted control signals into otherwise unused parts of the network message, in other words, a sequence of future control signals is transmitted through the network at each update time instead of single control signal [7]. There are also works in existence, which combine the ideas of the ETC and packet-based predictive control [8, 9, 10]. In [8] an event-triggered MPC setup for unconstrained systems was presented using input-to-state stability notions as basis, later extension of this work [9] proposed ETC schemes for constrained discrete time systems. Especially noteworthy is the fact that this extension utilizes optimal control sequences produced by MPC optimization problem in an open-loop manner between update times.

In this paper, an event-triggered control method is proposed, which aims at reducing computational complexity by further exploiting dynamic properties of the controlled system and known design parameters, effectively also reducing the complexity of design process when compared to some other predictive event-driven control strategies. Proposed method introduces modified control algorithm, which achieves specified goal, and a method for quantifying state divergence of two similar state-space models using preexistent design parameters. This approach yields an event-based event-triggered predictive optimal control method which reduces the need to solve optimization problem at each sampling period.

The following notations are used throughout this paper. The set of all real numbers $\mathbb{R}$ can be divided into its subsets by the notation shown below. Similar notation as for real numbers $\mathbb{R}$ can be used for other number sets *eg* integers $\mathbb{Z}$ and natural numbers $\mathbb{N}$. Using this notation, the set of all positive real numbers and zero can be denoted as $\mathbb{R}_{\geq 0}$. The set of natural numbers is defined as $\mathbb{Z}_{\geq 0} = \mathbb{N}$.

$$\mathbb{R}_{\square a} = \{x \in \mathbb{R} : x \square a\} \quad \forall a \in \mathbb{R}, \square \in \{>, <, \geq, \leq\} \,. \tag{1}$$

2 Problem formulation

State-space mathematical models comprised of first order vector difference equations, suitable for designing advanced control methods [2],

$$x(k+1) = Ax(k) + Bu(k)\,, \tag{2}$$

are considered in this paper. In (2) $k \in \mathbb{N}$ is number of the sample and corresponds to the time kT_S, T_S is a sampling period, $x(k) \in \mathbb{R}^n$ is a vector of discrete-time internal states, $u(k) \in \mathbb{R}^m$ is a vector of discrete-time control inputs, and $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ are matrices which specify the dynamic properties and behavior of the discrete-time model of a plant.

2.1 Model predictive control

The control strategy known as *Model Predictive Control* (MPC) or *Receding Horizon Control* (RHC) and *Moving Horizon Optimal Control* is a modern approach in control and aims at transformation of the control problem into an optimization one [11], which enables the optimization of the controlled plant behavior over controllable plant inputs $u(k)$ and predicted evolution of the plant state $\hat{x}(k+i|k)$.

Prediction of the plant state is obtained by utilizing an explicit numerical model of the controlled plant. Therefore the model is the essential element of an MPC controller. In the real world, the model is always imperfect estimation of the physical plant, thus the plant state forecast is never completely accurate. This inaccuracy can be partially overcome by implementing feedback from the output of the plant, [12].

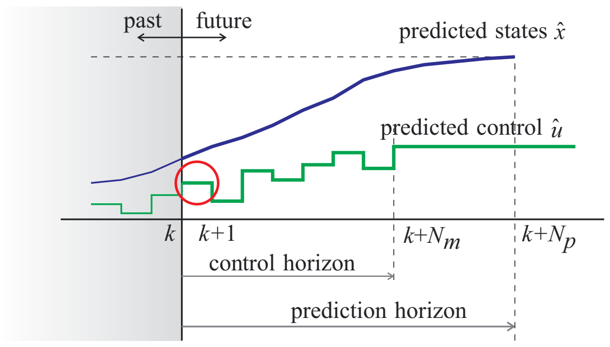

Fig. 1. Receding horizon strategy

An MPC controller is implemented by solving optimization problems the result is then applied according the *receding horizon* philosophy: At the instant k only the first optimal control output $\hat{u}(k|k)$ is actually applied to the plant. The remaining optimal control outputs are discarded and a new optimal control problem is solved at the instant $k+1$. This is illustrated in Fig. 1 and described by Algorithm 1

$$\begin{cases} \min_{u_{\cdot|k}} J(k, N_p, N_m, x(k), \hat{x}_{\cdot|k}, \hat{u}_{\cdot|k}) \\ \text{subject to} \\ x(k+1) = Ax(k) + Bu(k) \\ \text{“control and state constraints”} \\ \text{“stability constraints”}\,. \end{cases} \tag{3}$$

In (3) by $J(k, N_p, N_m, x(k), \hat{x}_{\cdot|k}, \hat{u}_{\cdot|k})$ is denoted a cost function, $\hat{x}_{\cdot|k}$ and $\hat{u}_{\cdot|k}$ the sequences of the predicted plant states and control outputs respectively were calculated at the time instants denoted as k after symbol $|$.

$$\hat{x}_{\cdot|k} = \big(\hat{x}(k+1|k), \hat{x}(k+2|k), \dots, \hat{x}(k+N_p|k)\big), \tag{4a}$$

$$\hat{u}_{\cdot|k} = \big(\hat{u}(k|k), \hat{u}(k+1|k), \dots, \hat{u}(k+N_m-1|k)\big). \tag{4b}$$

N_p denotes the length of the *prediction horizon* and N_m denotes the length of the *control horizon* ($N_m \leq N_p$). When $N_p = \infty$, we refer to this as the *infinite horizon problem*, and similarly, when N_p is finite as the *finite horizon problem* [13].

The cost function $J(k, N_p, N_m, x(k), \hat{x}_{\cdot|k}, \hat{u}_{\cdot|k})$ may have different forms depending on the optimization problem type (linear programming, quadratic programming, *etc*). Due to relative easy solution and availability of solvers, a receding horizon is often implemented using quadratic cost function of the form [13]

$$J(k, N_p, N_m, x(k), \hat{x}_{\cdot|k}, \hat{u}_{\cdot|k}) = \hat{x}^T(k+N_p|k)Q_0\hat{x}(k+N_p|k) + \sum_{i=0}^{N_p-1} \hat{x}^T(k+i|k)\,Q\,\hat{x}(k+i|k) + \sum_{i=0}^{N_m-1} \hat{u}^T(k+i|k)\,R\,\hat{u}(k+i|k)\,. \quad (5)$$

A basic MPC law is described by the following algorithm

Algorithm 1 – calculation of the basic MPC [3]

```
1. Get the new state $x(k)$.
2. Solve the optimization problem (3).
3. Apply $u(k) = \hat{u}(k|k)$.
4. $k \leftarrow k+1$. Go to 1.
```

2.2 Issues with model predictive control

2.2.1 Computational complexity

The computational complexity of the optimization problem‘s (3) solver is often of great concern. It depends on the internal workings of the solver, choice of the performance index (linear, quadratic (5), ...), state space and control outputs dimensions, lengths of the prediction and control horizons, *etc*.

2.2.2 Feasibility

In [14] authors have shown the feasibility of the optimal problem’s (3) solution at the initial time $k = 0$ does not necessary imply feasibility for all future times. It is desirable to design control strategy such that feasibility for all future times is guaranteed, a property which is called *persistent feasibility*.

Typically, feasibility is assumed at the time $k = 0$ and cost function and stability constrains (3) are chosen such that feasibility is preserved at the following time steps. This is achieved, for instance, by ensuring that shifted optimal sequence $\big(\hat{u}(k+1|k), \hat{u}(k+2|k), \ldots, \hat{u}(k+N_m-1|k), 0\big)$ is feasible at the time $k+1$. Also typically the constraints which impose restrictions on state variable in the optimization problem (3) can be treated as *soft* by adding a slack variable ϵ

$$G_2(\hat{x}_{\cdot|k}) \leq g_2 + \epsilon \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

The control output constraints $G_1(\hat{u}_{\cdot|k}) \leq g_1$ are maintained as *hard*. Relaxing the state constraints removes the feasibility problem at least for stable systems [13]. Keeping the state constraint hard does not make sense from the practical point of view because of the presence of the noise, disturbances and numerical errors. As the control outputs are yielded by the optimization procedure, control output constraints can always be regarded as hard [13].

2.2.3 Stability

To guarantee stability of the closed-loop system, additional constraints may be added to the optimization problem (3). There are several approaches to do this: *eg* the value $V(k) = J(k, N_p, N_m, x(k), \hat{x}_{\cdot|k}, u^*_{\cdot|k})$ obtained for the minimizer $u^*_{\cdot|k}$ of the form (4b) is used at each sample time k as the result of the *Lyapunov function*, other may include [13]:

End (terminal) constraint

Adds into (3) a stability constraint of the form [15]

$$\hat{x}(k+N_p|k) = 0\,. \quad (6)$$

Infinite prediction horizon

For asymptotically stable systems, no stability constraint is necessary when $N_p = +\infty$ [16, 17].

Terminal weighting matrix

When terminal matrix Q_0 in (5) is chosen as the solution of a *Riccati inequality*, stability can be guaranteed without the addition of any stability constraints [18].

Invariant terminal set

By relaxing terminal constraint (6) into set membership constraint

$$\hat{x}(k+N_p|k) \in \Omega,$$

and introducing an LQ feedback gain F_{LQ}.

$$\hat{u}(k+i|k) = F_{LQ}\hat{x}(k+i|k), \forall i \geq N_m\,.$$

The set Ω is an *invariant set* (*ie* a set of states $x(k)$ which, once entered by the system, will be never left [14]) under LQ regulation and the constraints are met inside Ω [19].

Contraction constraint

Instead of the terminal cost $V(k)$ as a value of the Lyapunov function, explicitly requires some special property of the state $\hat{x}(k+i|k)$. To ensure stability, this property might be that some norm of the state decreases with time [20]

$$||\hat{x}(k+i|k)|| \leq \alpha||x(k)||, \forall i \in \{1, \ldots, N_p\}.$$

3 Event-driven model predictive control

In this section, the modified control method is proposed. This method is aimed at reducing computational complexity of an MPC method by exploiting preexistent control signals and design parameters.

Result of the optimization process are sequence of control signals (4b) and a sequence of predicted state variables (4a)

$$\begin{cases} \min_{u_{\cdot|k}} J(k, N_p, N_m, x(k), \hat{x}_{\cdot|k}, \hat{u}_{\cdot|k}) \\ \text{subject to} \\ x(k+1) = Ax(k) + Bu(k)\,, \\ G_1 \widetilde{u}(\cdot|k) \leq g_1\,, \\ G_2 \widetilde{x}(\cdot|k) \leq g_2\,. \end{cases} \tag{7}$$

$$\widetilde{x}(\cdot|k) = \begin{bmatrix} \hat{x}(k+1|k) \\ \vdots \\ \hat{x}(k+N_p|k) \end{bmatrix}, \tag{8a}$$

$$\widetilde{u}(\cdot|k) = \begin{bmatrix} \hat{u}(k|k) \\ \vdots \\ \hat{u}(k+N_m-1|k) \end{bmatrix}. \tag{8b}$$

In (7), $J(k, N_p, N_m, x(k), \hat{x}_{\cdot|k}, \hat{u}_{\cdot|k})$ is a cost function, such as (5), G_1, g_1 and G_2, g_2 are parameters which specify the control and state constraints respectively, $\widetilde{x}(\cdot|k)$ and $\widetilde{u}(\cdot|k)$ are vectors of predicted state (8a) and control (8b) signals respectively.

From 2.1 it follows that, in standard MPC, the optimization problem (7) is solved at each sampling period k and only the first control signal of the sequence (4b) is used. The method proposed here is based on the assumption that the control signals sequence (4b) can be utilized throughout control horizon N_m in an open-loop manner, provided the predicted state $\hat{x}(k+i|k)$ does not divert from the measured state $x(k+i)$ for all $i \in \{1, 2, \ldots, N_m - 1\}$. This assumption can be expected to hold, as shown in (9), for limited number of sample periods $l \in \mathbb{N}$, even in real-world scenarios, where bounded unmeasurable disturbance or model inaccuracy is affecting system behavior

$$\exists\, l \in \{0, \ldots, N_m - 1\}\,,\ \big(\hat{x}(k+i|k) - x(k+i)\big) \in \epsilon_0 \quad \forall i \in \{0, \ldots, l\}\,, \tag{9}$$

where $\epsilon_0 \subset \mathbb{R}^n$ represents subspace of the state space which consists of the origin and its arbitrary small neighborhood, $\hat{x}(k+i|k) \in \mathbb{R}^n$ and $x(k+i) \in \mathbb{R}^n$ are current state of the plant predicted at the time instant k and current actual state of the plant respectively.

The proposed method can be used to trade between precision of the control and the computational resources needed by reducing the number of optimization problems solved over some time horizon using (9) and

$$u(k+i) = \hat{u}(k+i|k)\ \forall i \in \{0, \ldots, l\}\,, \tag{10}$$

where l refers to the variable defined in (9).

3.1 Predicted state divergence

Continuous-time LTI model eqrefeq:disturbed-linear-continuous-plant reflects uncertainties in the model of a real plant by introducing the disturbance into model‘s dynamics.

$$\dot{x}'_c(t) = A_c x'_c(t) + B_c u_c(t) + W_c w_c(t)\,, \tag{11}$$

where $w_c(t) \in \mathbb{R}^q$ is a disturbance input and $W_c \in \mathbb{R}^{n\times q}$ is a disturbance coupling matrix. In real world the disturbed model properties $w_c(t)$ and W_c are usually unmeasurable and unknown respectively and therefore do not appear in the *design model* (12). The unawareness of the precise model during the design is usually overcome by implementing some robustness into applied control method.

$$\dot{x}_c(t) = A_c x_c(t) + B_c u_c(t)\,. \tag{12}$$

By comparing continuous disturbed model (11) and continuous design model (12) it is easy to see that the respective state variables of these models might diverge with time $t \in (t_0, \infty)$, when $x'_c(t_0) = x_c(t_0)$, $W_c \neq 0_{n\times q}$ and $\exists\, t \in (t_0, \infty)$ so that $w_c(t) \neq 0_q$.This introduces difference variable $e(k+i|k) \in \mathbb{R}^n$ defined by

$$e(k+i|k) = x(k+i) - \hat{x}(k+i|k)\ \forall i \in \{0, \ldots, N_m - 1\}\,. \tag{13}$$

When employing an MPC, where solution of the optimization problem is event-triggered [6], a predicted state divergence can be calculated from predicted state $\hat{x}(k+i|k)$ and an actual measured state $x(k+i)$, where $k+i$ is current time instant and k is the time instant of prediction calculation.

The proposed method for quantifying the state divergence is a vector norm $||\cdot||\colon \mathbb{R}^{n\times 2} \to \mathbb{R}_{\geq 0}$ (14) of the difference between measured state and the state predicted at the instant k, where $x(k+i) \in \mathbb{R}^n$, $\hat{x}(k+i|k) \in \mathbb{R}^n$, $k \in \mathbb{N}$ and $i \in \mathbb{N}$.

$$||e(k+i|k)|| = ||x(k+i) - \hat{x}(k+i|k)||\ \forall i \in \{1, \ldots, N_m - 1\}\,. \tag{14}$$

The difference norm can be implemented using the preexistent parameters from the MPC cost function (5), which yields quadratic cost of difference denoted by $J_e\colon \mathbb{R}^{n\times 2} \to \mathbb{R}_{\geq 0}$. This way the Q parameter from the MPC quadratic cost function (5) is reused in a compelling manner, removing the need for adding more design variables. Such norm of the difference is presented in (15) which also handles the boundaries of the control horizon N_m.

$$J_e(x(k+i), \hat{x}(k+i|k)) = \begin{cases} 0 & \text{for } i = 0\,, \\ e^T Q e & \text{for } 0 < i < N_m\,, \\ \infty & \text{for } i \geq N_m\,, \end{cases} \tag{15}$$

where $e \in \mathbb{R}^n$ is short for $e(k+i|k)$.

3.2 Modified control approach

Since this method is based on an on-line optimal control strategy, it is essential to first design such control. Proposed modified method is based on the MPC control approach reviewed in section 2.1. The proposed approach also introduces some new design concerns, first of which is the question how to measure divergence of the predicted state $\hat{x}(k+i|k)$ from the real measured state $x(k+i)$. This questions was tackled in section 3.1 and here the cost of state difference $J_e(x(k+i), \hat{x}(k+i|k))$ (15) is used.

The second design concern is the threshold value of the calculated difference norm. This value $||e||_{\text{threshold}} \in \mathbb{R}_{\geq 0}$ is positive real number or zero which limits the value of the difference cost $||e||$ (implemented in proposed method by difference cost J_e).

Algorithm 2 Event-driven approach to model predictive control

```
1. Obtain $x(k)$
2. loop
3.     Solve the optimization problem (7)
       and obtain sequences $\hat{x}_{\cdot|k}$ and $\hat{u}_{\cdot|k}$ (4)
4.     $i \leftarrow 0$
5.     while $J_e(x(k+i), \hat{x}(k+i|k)) \leq ||e||_{\text{threshold}}$ do
6.         Apply $u(k+i) = \hat{u}(k+i|k)$
7.         $i \leftarrow i+1$
8.         Obtain $x(k+i)$
9.     end while
10.    $k \leftarrow k+i$
11.end loop
```

The Complete algorithmic description of the proposed modified method is shown in Algorithm 2. On the lines 1 and 8 of Algorithm 2 the current state information $x(k)$ is obtained, which is done either by directly measuring the state, when it is available, or by recovering it with some of the state observing methods. The optimization problem yielding the control signals is solved on line 3. These control signals are used on the lines 5 to 9. On line 5 the difference cost J_e (15) is evaluated and control signals are used only in case this value does not exceed specified threshold $||e||_{\text{threshold}}$.

From Algorithm 2, it is evident that setting the threshold value $||e||_{\text{threshold}}$ above zero might lead to sparing computational resources by reusing the existing control signals inside the predicted control horizon N_m. This will cause the most computationally demanding line of the Algorithm 2 (line 3) to be executed less often. On the other hand, by setting higher threshold value $||e||_{\text{threshold}}$, a greater divergence from the predicted state is allowed. This may result in lower accuracy of the control. Therefore, it is possible to gather that changing the threshold value $||e||_{\text{threshold}}$ allows to trade between the accuracy of the control and computational resources used by the control algorithm.

It can be easily seen, that this method does not discard stability, for $||e||_{\text{threshold}} < \infty$, in case the control method on which it is built on (MPC) guarantees stability. This is due to the fact, that the $J_e(x(k+i), \hat{x}(k+i|k))$ function can generally be defined as a *Lyapunov function*, limiting the amount of (generalized) energy of the difference between predicted and actual state of the system.

In its core the method proposed here limits the value of the quadratic form (16), defined by (15), to the maximum of $||e||_{\text{threshold}}$

$$e(k+i|k)^\top Q e(k+i|k) \,. \tag{16}$$

A *contracted constraints* need to be calculated which when used by optimization algorithm (line 3 in Algorithm 2) ensure state constraints, introduced by (7), are being held.

A conservative approach to do such constraint contraction can be proposed using maximum divergence of each state variable $e_{\max}(Q, ||e||_{\text{threshold}}) \in \mathbb{R}^n$ allowed by Q and$||e||_{\text{threshold}}$ parameters. It is safe to assume that matrix Q is positive definite, since quadratic form (16) origins form cost function of MPC control law (5). This fact can be exploited to obtain the maximum divergence of each state variable $e_{\max}$. Form the geometrical point of view, this means that (16) defines an ellipsoid in $\mathbb{R}^n$ space and $e_{\max}$ describes its *axis-aligned minimum bounding box*. This bounding box can be obtained using a method described in [21].

Conceivably this approach places upper bound limit on the $||e||_{\text{threshold}}$ value, since too big $||e||_{\text{threshold}}$ would render optimization problem in Algorithm 2 unfeasible for all $x(k) \in \mathbb{R}^n$.

4 Results

To present implications of the theory proposed in this paper following subsections demonstrate its usage on a simple model inside a simulated environment. The first subsection shortly describes the transformation of a MPC with quadratic cost function into a general quadratic optimization problem. Next a simulation model and its parameters are presented. Last subsection shortly summarizes results of the simulated experiment.

4.1 Control Design

An MPC optimization problem (7) with quadratic cost function is to be transformed here into general quadratic programming optimization problem (17) as described in [14]. This form is usually used by generic solvers of optimization problems with a quadratic cost function.

$$\begin{cases} \min_z \frac{1}{2} z^\top H z + q^\top z + r \\ \text{subject to} \\ G_i z \leq w_i \,, \\ G_e z = w_e \,. \end{cases} \tag{17}$$

The optimization problem (17) is a generic quadratic optimization problem with both inequality constraints and equality constraints, where $z \in \mathbb{R}^s$, $H \in \mathbb{R}^{s\times s}$ is a symmetric ($H = H^\top$) positive definite matrix, $q \in \mathbb{R}^s$ and $r \in \mathbb{R}$. Constraints are comprised of inequality constraints defined by $G_i \in \mathbb{R}^{n_i\times s}$ and $w_i \in \mathbb{R}^{n_i}$ and equality constraints defined by $G_e \in \mathbb{R}^{n_e\times s}$ and $w_e \in \mathbb{R}^{n_e}$.

4.1.1 Cost function formulation

First the optimization vector z is derived as the concatenation of the optimization sequences (4a) and (4b)

$$z(k) = \begin{bmatrix} x(k) \\ \widetilde{x}(\cdot|k) \\ \widetilde{u}(\cdot|k) \end{bmatrix}. \tag{18}$$

In (18) $\widetilde{x}(\cdot|k)$ and $\widetilde{u}(\cdot|k)$ are vectors of predicted states and control output defined in (8a) and (8b) respectively.

Further, from (5) the MPC cost function can be derived a quadratic cost matrix H. Easily can be seen that neither linear (q) nor constant (r) component is present in a MPC cost function (5).

By considering vector z defined as in (18), the matrix H can be derived into form

$$H = \mathrm{Diag}\left(Q, \ldots, Q, Q_0, R, \ldots, R\right), \tag{19}$$

where the counts of Q and R matrices are equal to the lengths of the prediction horizon N_p and control horizon N_m respectively.

4.1.2 Constraints formulation

Inequality constraints comprised of a matrix G_i and a vector w_i are easily acquired from the limitations acting upon state variables and control signals of the system which will be presented along the model of the plant in section 4.2.

In an MPC optimization problem (7) the only equality constraints are formed by the system dynamics equation (2). To obtain the matrix G_e and the vector w_e, first an augmented system dynamics equation (20) is to be derived

$$\widetilde{x}(\cdot|k) = \widetilde{A}x(k) + \widetilde{B}\widetilde{u}(\cdot|k), \tag{20}$$

where

$$\widetilde{A} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{N_p} \end{bmatrix}, \tag{21}$$

$$\widetilde{B} = \begin{bmatrix} B & 0_{n\times m} & \cdots & 0_{n\times m} \\ AB & B & \cdots & 0_{n\times m} \\ \vdots & \vdots & \ddots & \vdots \\ A^{N_m-1}B & A^{N_m-2}B & \cdots & B \\ \vdots & \vdots & \ddots & \vdots \\ A^{N_p-1}B & A^{N_p-2}B & \cdots & \sum_{i=N_m}^{N_p} A^{N_p-i}B \end{bmatrix}. \tag{22}$$

Using the matrices $\widetilde{A}$ and $\widetilde{B}$, the equality constraints are defined as

$$\begin{aligned} G_e &= \begin{bmatrix} I_{n\times n} & 0_{n\times(n\cdot N_p)} & 0_{n\times(m\cdot N_m)} \\ \widetilde{A} & -I_{(n\cdot N_p)\times(n\cdot N_p)} & \widetilde{B} \end{bmatrix}, \\ w_e(k) &= \begin{bmatrix} x(k) \\ 0_{(n\cdot N_p)\times 1} \end{bmatrix}. \end{aligned} \tag{23}$$

4.2 Model

Model used to present the theory (Fig. 2), representing a generalized mechanical suspension, consists of several masses m_i interconnected by combination of springs and dampers. Each such mass can be described by differential equations

$$\begin{aligned} \dot{x}_{c,i_1} &= x_{c,i_2}, \\ \dot{x}_{c,i_2} &= \frac{1}{m_i}\sum_{j\in N_i}\Big[-k_{i,j}\big(x_{c,i_1}-x_{c,j_1}\big)- \\ &\qquad b_{i,j}\big(x_{c,i_2}-x_{c,j_2}\big)\Big] + \frac{100u_{c,i_1}}{m_i}, \\ \dot{x}_{c,i_3} &= x_{c,i_4}, \\ \dot{x}_{c,i_4} &= \frac{1}{m_i}\sum_{j\in N_i}\Big[-k_{i,j}\big(x_{c,i_3}-x_{c,j_3}\big)- \\ &\qquad b_{i,j}\big(x_{c,i_4}-x_{c,j_4}\big)\Big] + \frac{100u_{c,i_2}}{m_i}, \end{aligned} \tag{24}$$

where $i \in \mathcal{M} \subset \mathbb{Z}_{>0}$ is an index of the mass, m_i is a weight of a mass, $k_{i,j} = k_{j,i}$ and $b_{i,j} = b_{j,i}$ are a spring constant and a damping ration respectively between mass i and mass j, x_{c,i_1} and x_{c,i_3} are vertical and horizontal displacement from the equilibrium respectively, x_{c,i_2} and x_{c,i_4} are the vertical and horizontal velocities respectively, $100u_{c,i_1}$ and $100u_{c,i_2}$ are the vertical and horizontal forces respectively applied to mass i, $\mathcal{N}_i \subset \mathcal{M}$ is a set of masses interconnected with the mass i.

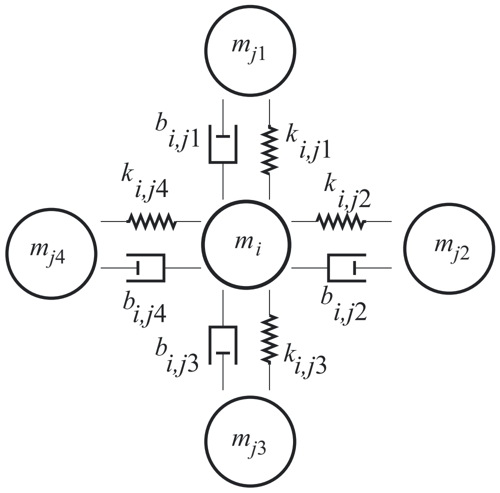

Fig. 2. Mechanical system

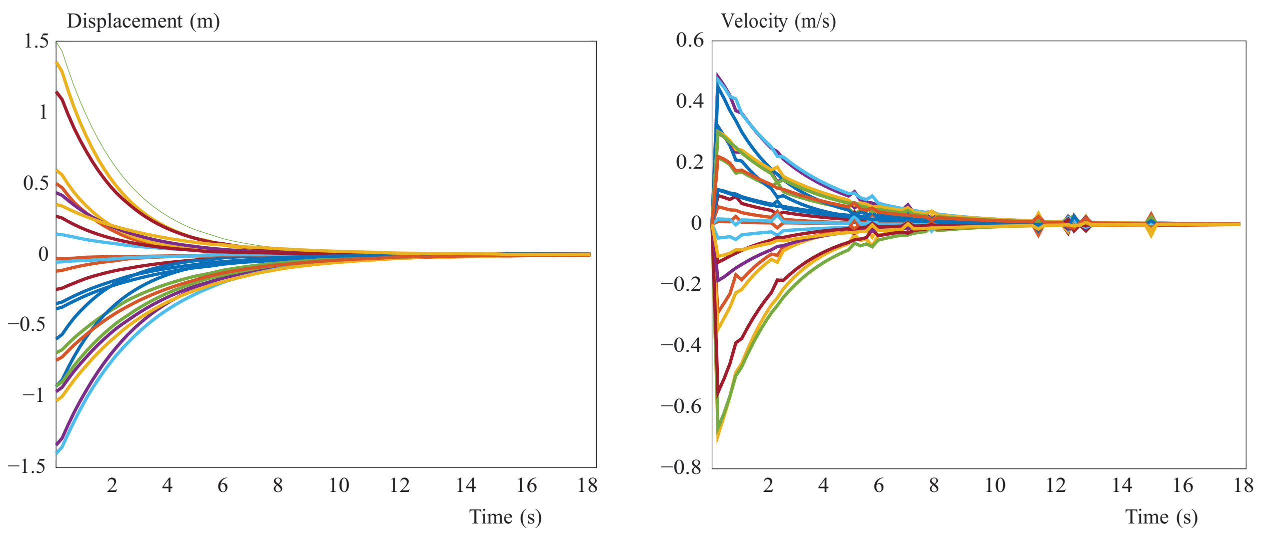

Fig. 3. Displacement state variables, Velocity state variables, Evolution of the mechanical system controlled by MPC

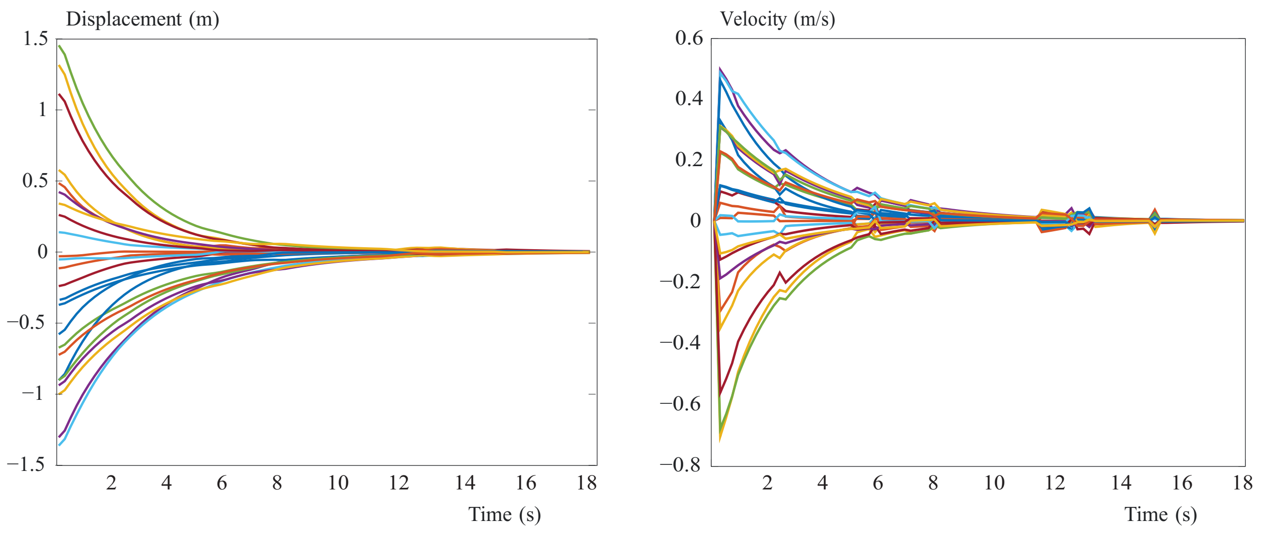

Fig. 4. Displacement state variables, Velocity state variables, Evolution of the mechanical system controlled by MPC

To achieve reasonably realistic results, state variables and inputs of each subsystem, composed by a single mass, are constrained. These constraints are shown in Table 1 and apply to all masses.

Table 1. Constraints of the plant

Variable	Maximum absolute value
x_{c,i_1}	$1.5\,\mathrm{m}$
x_{c,i_2}	$0.8\,\mathrm{m\,s^{-1}}$
x_{c,i_3}	$1.5\,\mathrm{m}$
x_{c,i_4}	$0.8\,\mathrm{m\,s^{-1}}$
$100u_{c,i_1}$	$125\,\mathrm{N}$
$100u_{c,i_2}$	$125\,\mathrm{N}$

Overall model of the designed system (11) consists of 3×4 interconnected masses. Initially, the masses are randomly displaced from their respective equilibria inside the subspace of the state-space defined by constraints shown in Table 1. Also a disturbance signal $w_c(t) \in \mathbb{R}^q$ is acting upon the model in random time instants.

In order to properly exploit the knowledge of the physical properties of the designed mechanical system the weight matrix Q in the MPC cost function (5) can be designed to reflect the total energy of the system, as illustrated by

$$Q = \mathrm{Diag}\left(\sum_{\mathrm{V}} k_{i,j}, m_i, \sum_{\mathrm{H}} k_{i,j}, m_i, \dots\right), \qquad (25)$$

where $\sum_{\mathrm{V}} k_{i,j}$ and $\sum_{\mathrm{H}} k_{i,j}$ are sums of constants of springs attached to the mass i in vertical respectively in horizontal direction.

5 Discussion

After construction and discretization of the model described above, two control strategies were designed: a

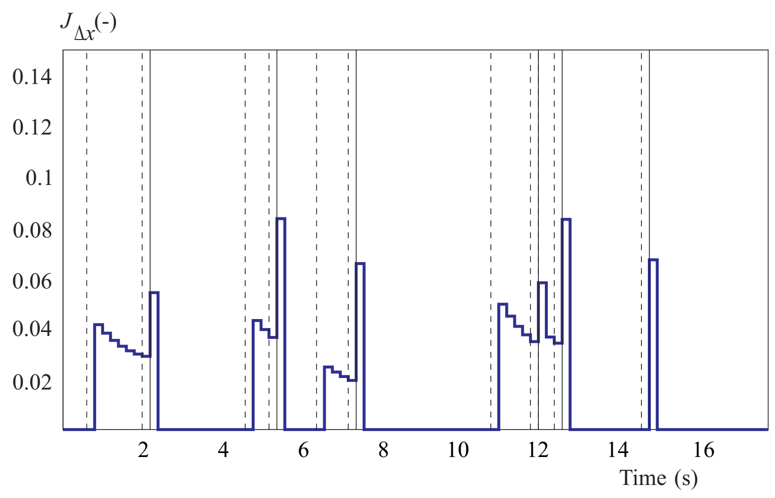

Fig. 5. Evolution of the difference cost J_e

classical MPC approach and the method outlined in this paper.

The results of the simulated control of the model by MPC are shown in Fig. 3, which displays displacement and velocity evolution of the model. The same state evolutions of the simulated system controlled by the method proposed in this paper are shown in Fig. 4. For the latter of the simulations, also an evolution of the difference cost J_e (15) is shown in Fig. 5. Mentioned figures clearly show that both control strategies successfully steer the system into origin of its state-space, effectively achieving the goal of the control set by the cost function (5).

The first simulation, resulting in state evolutions shown in Fig. 3, exemplifies MPC method applied on the model described in section 4.2. The second simulation, presented by Figs. 4 and 5, is done on the same model with the same initial conditions $x(0)$ and same noise signal $w(k)$ acting upon it as the first simulation. The difference is the control strategy, which is the method proposed in this paper.

The main aim of the proposed approach is to reduce computational expensiveness of the MPC. Presented simulations suggest that this goal is met, since the first approach runs the optimization problem solver (3) once per each sampling period, *ie* 90 *times* during the presented simulation. From Fig. 5 it is obvious that the modified approach executes the same optimization problem solver only 7 *times* (including the first run at time 0).

On the other hand, by comparing state evolutions of the system controlled by MPC and by the method proposed in this paper it is apparent that the latter shows noticeable deviations from the optimal trajectory presented by the former. This are caused by non-zero threshold $J_{\text{threshold}}$ (see line 5 in Algorithm 2. This results seem to confirm the proposition that $J_{\text{threshold}}$ can be used to trade between precision of the control and computational resources need by the control algorithm.

6 Conclusion

This paper provides, in its first part, a brief review of MPC approach to controlling dynamic systems and continues by proposing new method aiming at minimizing the computational resources needed by this control method while reasonably and in a bounded way reducing the precision of the control. Experimental simulated results are presented in the second part of this paper. As mentioned in section 5, this section describes and provides comparison of two simulated scenarios done using a model described therein. First scenario presents the standard MPC and second simulation shows behavior of the method proposed in this paper. The comparison clearly shows the reduced usage of the computation resources by reducing the number of optimization problems solved over simulated time period.

Contributions: Stefan Misik proposed the methods, performed experiments and partially wrote the paper. As the co-author of the paper, Zdenek Bradac played an important role during the model and simulation design and partially wrote the paper. Arben Cela played an important role in designing the methods used in this paper and also revised the paper and provided many suggestions. All authors read and approved the final manuscript.

Acknowledgements

The completion of this paper was made possible by the grant No. FEKT-S-17-4234 – "Industry 4. 0 in automation and cybernetics" financially supported by the Internal science fund of Brno University of Technology.

References

[1] K. J Åström and B. Wittenmark, *Computer-Controlled Systems*, Prentice Hall, Upper Saddle River, New Jersey 07458, 3rd edition, 1997.

[2] F. L. Lewis, *Applied optimal control*, Prentice Hall, Englewood Cliffs, N.J., 1992.

[3] M. E. M. B. Gaid, A. Cela, and Y. Hamam, "Optimal Integrated Control and Scheduling of Networked Control Systems

with Communication Constraints: Application to a Car Suspension System", *IEEE Transactions on Control Systems Technology*, vol. 14, no. 4, pp. 776–787, July 2006.

[4] M. E.-M. B. Gaid, "Optimal Scheduling and Control for Distributed Real-Time Systems", *PhD thesis*, Université d'Evry Val d'Essonne, November 2006.

[5] T. M. P. Gommans and W. P. M. H. Heemels, "Resource-Aware MPC for Constrained Nonlinear Systems: A Self-Triggered Control Approach", *Systems & Control Letters*, vol. 79, pp. 59–67, 2015.

[6] W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada, "An Introduction to Event-Triggered and Self-Triggered Control", *2012 IEEE 51-st IEEE Conference on Decision and Control (CDC)*, pp. 3270–3285, December 2012.

[7] L. Greco, A. Chaillet and A. Bicchi, "Exploiting Packet Size Uncertain Nonlinear Networked Control Systems", *Automatica*, vol. 48, no. 11, pp. 2801–2811, Nov 2012.

[8] A. Eqtami, D. V. Dimarogonas and K. J. Kyriakopoulos, "Event-Triggered Control for Discrete-Time Systems", *Proceedings of the 2010 American Control Conference. Institute of Electrical and Electronics Engineers (IEEE)*, 1.6.2010.

[9] A. Eqtami, D. V. Dimarogonas and K. J. Kyriakopoulos, "Event-Triggered Strategies for Decentralized Model Predictive Controllers", *IFAC Proceedings*, vol. 44, no. 1, pp. 10068–10073, Jan 2011.

[10] D. Bernardini and A. Bemporad, "Energy-Aware Robust Model Predictive Control based on Noisy Wireless Sensors", *Automatica*, vol. 48, no. 1, pp. 36–44, Jan 2012.

[11] R. Scattolini, "Architectures for Distributed and Hierarchical Model Predictive Control – a Review", *Journal of Process Control*, vol. 19, no. 5, pp. 723–731, 2009.

[12] J. B. Rawlings, "Tutorial Overview of Model Predictive Control", *Control Systems*, IEEE, vol. 20, no. 3, pp. 38–52, June 2000.

[13] A. Bemporad and M. Morari, "Robust Model Predictive Control: A Survey", A. Garulli and A. Tesi, editors, *Robustness Identification and Control*, vol. 245 of Lecture Notes Control and Information Sciences, pp. 207–226, Springer London, 1999.

[14] F. Borrelli, A. Bemporad and M. Morari, *Predictive Control*, [Unpublished], 1.10.2015.

[15] W. H. Kwon and A. E. Pearson, "A Modified Quadratic Cost Problem and Feedback Stabilization of a Linear System", *Automatic Control*, IEEE Transactions on, vol. 22, no. 5, pp. 838–842, Oct 1977.

[16] S. S. Keerthi and E. G. Gilbert, "Optimal Infinite-Horizon Feedback Control Laws for a General Class of Constrained Discrete-Time Systems: Stability and Moving-Horizon Approximations", *Journal of Optimization Theory and Applications*, vol. 57, no. 2, pp. 265–293, May 1988.

[17] J. B. Rawlings and K. R. Muske, "The Stability of Constrained Receding Horizon Control", *Automatic Control, IEEE Transactions on*, vol. 38, no. 10, pp. 1512–1516, Oct 1993.

[18] W. H. Kwon, A. M. Bruckstein and T. Kailath, "Stabilizing State-Feedback Design via the Moving Horizon Method", In *Decision and Control*, 1982 21st IEEE Conference on, pp. 234–239, Dec 1982.

[19] P. O. M. Scokaert and J. B. Rawlings, "Infinite Horizon Linear Quadratic Control with Constrains", *IFAC*, vol. 7a-04 no. 1, pp. 109–114, San Francisco, USA, 1996.

[20] E. Polak and T. H. Yang, "Moving Horizon Control of Linear Systems with Input Saturation and Plant Uncertainty Part 1. Robustness", *International Journal of Control*, vol. 58, no. 3, pp. 613–638, 1993.

[21] F. Domes and A. Neumaier, "Rigorous Enclosures of Ellipsoids and Directed Cholesky Factorizations", *SIAM Journal on Matrix Analysis and Applications*, vol. 32, no. 1, pp. 262–285, Jan 2011.

Received 2 February 2017

Štefan Misík was born in 1990. Received engineering degree MSc in cybernetics from Brno University of Technology, Czech Republic in 2014. Currently, he works on his PhD at Department of Control and Instrumentation, Faculty of Electrical Engineering and Communication, Brno University of Technology, Czech Republic.

Zdeněk Bradáč was born in 1973. He received his PhD in technical cybernetics in 2004 at Brno University of Technology. His research interests include HMI systems, fault-tolerant systems, information systems safety and security. He is an associate professor at Department of Control and Instrumentation, Faculty of Electrical Engineering and Communication, Brno University of Technology.

Arben Cela received the engineering diploma from the Polytechnic University of Tirana, Albania in 1984. From 1984 to 1986 he was with the Department of Power Generation of Faculty of Engineering of Polytechnic University of Tirana, Albania. He received the PhD and HDR diploma from Universit Paris-Sud respectively in January 1993 and June 2011. From 2000 to 2012 he was the head of Embedded Systems Department of the ESIEE Paris, Noisy-Le-Grand France. From June 2012 he is with Computer Science and Telecommunication department of ESIEE Paris. Recently, he has participated to the scientific committee of IFAC Workshop PDeS (2006, 2009, 2012), 2nd IFAC Workshop on Distributed Estimation and Control in Networked Systems (2010), IEEE International Conference on System Theory and Control 2010, and Internationale Francophone Control Conference 2012. He is author of more than 80 conference and journal papers in the area of optimization, control of distributed real time systems and delayed systems.