PRE-STABILIZED ENERGY-OPTIMAL MODEL PREDICTIVE CONTROL

Xin Wang  
Department of Mechanical Engineering, KU Leuven,  
Mailbox 2420, Celestijnenlaan 300B, B-3001 Heverlee, Belgium  
email: xin.wang@mech.kuleuven.be

Julian Stoev  
Flanders’ Mechatronics Technology Centre, Celestijnenlaan 300 D,  
3001 Leuven, Belgium  
email:julian.stoev@fmtc.be

Jan Swevers  
Department of Mechanical Engineering, KU Leuven, Mailbox 2420, Celestijnenlaan 300B, B-3001 Heverlee, Belgium  
email: Jan.Swevers@mech.kuleuven.be

ABSTRACT
This paper presents Pre-stabilized Energy-optimal Model Predictive Control which is developed based on the existing Energy-Optimal Model Predictive Control (EOMPC) approach. EOMPC is a control method to realize energy-optimal point-to-point motions within a required motion time. In order to obtain a sufficiently large prediction time horizon with a limited number of decision variables resulting in less computational load and solving the optimization problem within the chosen sampling time, non-equidistant time intervals are used over the prediction horizon. This approach is called blocking. However blocking yields a non-smooth optimal solution and as a result the energy-optimality is only approximately achieved. In order to overcome this drawback, this paper proposes a pre-stabilization strategy to reduce the computational load of EOMPC. Pre-stabilization uses deadbeat state feedback to modify the system models employed in the formulation of MPC and yields a much sparser optimization problem. The significant advantage of the pre-stabilization on computational speed of MPC optimization problems is clarified. The computational efficiency and performance of EOMPC with pre-stabilization is validated through numerical simulations.

KEY WORDS
Motion control, Model based control, Energy optimal control, Embedded mechanical systems.

1 Introduction

Most mechatronic systems are controlled using linear feedback controllers such as traditional PID controllers [1] and this is mainly because these linear controllers are easy to implement and tune. Their main disadvantage is that they cannot explicitly take system constraints into account, and are hence less suited for applications with time/energy optimality requirements. Model predictive control (MPC) [2][3] is a much more appropriate control approach for these optimal control problems, and due to recent advances in fast numerical solution methods [4], the MPC approach is becoming more and more popular in mechatronic applications where sampling rates are much higher than in process control applications where MPC was introduced first [5]. MPC algorithms calculate future control actions by solving at each sampling time an optimization problem specified over a certain prediction horizon for a given system model, a given current system state and reference signal, and taking into account constraints on inputs, outputs and states.

Recently a MPC algorithm called Energy-Optimal Model Predictive Control (EOMPC) [6] has been developed based on the Time-Optimal Model Predictive Control (TOMPC) approach [7] which is a method to realize time-optimal point-to-point motion control. EOMPC aims at performing energy-optimal point-to-point motions within a required motion time. Energy optimality is achieved by setting the object function of the MPC optimization problem equal to the system’s energy consumption. An application of the EOMPC approach on a badminton robot described in [8] shows its practical applicability. The EOMPC optimization problem is a convex quadratic program (QP) and the size of which dependent on the number of decision variables. The number of decision variables depends on the length of the prediction horizon which has to be limited in order to solve the optimization problem within the chosen sampling time. However, if large point-to-point motions have to be performed, the total prediction time have to be sufficiently large in order to have a feasible solution. Non-equidistant time intervals over the prediction horizon are introduced such that a sufficiently large prediction time horizon can be achieved with a limited number of decision variables. This approach is called ‘blocking’ [8]. However blocking yields non-smooth optimal solutions and as a result energy-optimality is achieved only approximately.

This paper presents the Pre-stabilized Energy-optimal Model Predictive Control (Pre-stabilized EOMPC) strategy which is developed based on EOMPC. In the Pre-stabilized EOMPC, instead of using the blocking strategy, we utilize the ’pre-stabilization’ strategy to reduce the computational load of the EOMPC. The idea of pre-stabilization comes from the literature on Generalized Predictive Control (GPC) [9]-[11], where it was introduced to modify the open-loop optimal control problem employed in model predictive control of constrained systems so that closed-loop stability could be guaranteed. [12] shows that the pre-stabilization has significant computational and numerical advantages for open-loop unstable systems. In [13]
and [14] this idea was extended to state-space models and deadbeat state feedback and that leads to improvements in numerical conditioning. Here we show that the pre-stabilization also has significant advantage on computational speed of MPC optimization problems for both stable/unstable systems. In this paper, pre-stabilization is used in EOMPC in order to obtain a sparse optimization problem such that computational load is much less dependent on the number of decision variables. Numerical simulation presented in this paper confirms this statement.

**Outline of the paper:** The paper starts with a brief description of the EOMPC approach. Then the pre-stabilized EOMPC approach is explained in section 3. Section 4 discusses the numerical validation of the developed approach. In addition, a comparison of EOMPC and pre-stabilized EOMPC is provided. The conclusions are drawn in the last section.

## 2 Energy-Optimal MPC

In this section, the basic formulation of EOMPC is taken from our own previous work [8] and is repeated here for convenience. EOMPC is a MPC approach for linear time-invariant (LTI) systems aiming at performing energy-optimal point-to-point motions within a required motion time $T$ [8] [6]. The EOMPC approach determines the control signal by solving on-line, at every sampling time, an optimal control problem, based on the current state of the open-loop system model as shown in Fig. 1. In the EOMPC approach the settling time is defined as the number $N$ of discrete time sampling instants required for the system to be at rest at the desired set point. In order to guarantee the motion time, the settling time of the system is minimized until the requested motion time $T$ is reached. It is assumed that $T$ is integer multiple $K^*$ of the sampling time $T_s$: $T = K^* T_s$. Energy optimality is achieved by setting the object function of the MPC optimization problem equal to the system’s energy consumption $E_{loss}$.

### Figure 1. The EOMPC approach is based on an open-loop system model

The EOMPC optimization problem is formulated as a two-layer optimization problem [8]. The top layer is called *Problem B* and determines the settling time. Finding the settling time involves solving a series of feasibility problems. This feasibility problem is the second layer and is called *Problem A*. The optimal control sequence is the last feasible solution of *Problem A*. The following paragraphs describe *Problem A* and *Problem B* in detail.

*Problem A* denoted as $P_A(\hat{x}_t, N)$ is defined in Eq. (1). It calculates the energy optimal control signal for a given $N$ (which is obtained by solving *Problem B*) while respecting the system constraints.

$$V^* = \min_{u} E_{loss}(u)$$

s.t. 

$$x_0 = \hat{x}_t$$

$$x = X x_0 + X_u u,$$ 

$$y = Y x_0 + Y_u u,$$

$$e_1 \leq H x + G u \leq e_2,$$

$$y_k = y_{ref}, \quad k = N, \ldots, N + n - 1$$

$$u_k = 0, \quad k = N, \ldots, N_{max} - 1$$

In Equations (1), $u = [u_0, \ldots, u_{N_{max} - 1}]^T$ is the decision variable over the prediction horizon $N_{max}$. $E_{loss}(u)$ is the energy consumption of the system as a function of $u$ and is minimized in Eq. (1a). $\hat{x}_t$ in Eq. (1b) is the system state at time instant $t$. Based on the open-loop discrete-time state-space system model, the system state sequence $x = [x_0^T, \ldots, x_{N_{max}}^T]^T$ and the system output sequence $y = [y_0, \ldots, y_{N_{max}}]$ over the prediction horizon are defined in Eq. (1c) and (1d). $x$ and $y$ depend on the current state $x_0$ and the decision variable $u$ with matrices $X_x$, $X_u$, $Y_x$ and $Y_u$ defined as follows:

$$X_x = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{N_{max}} \end{bmatrix}$$

$$X_u = \begin{bmatrix} B & 0 & \cdots & 0 \\ AB & B & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{N_{max} - 1} B & A^{N_{max} - 2} B & \cdots & B \end{bmatrix}$$

$$Y_x = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{N_{max}} \end{bmatrix}$$

$$Y_u = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ C A^{N_{max} - 1} B & C A^{N_{max} - 2} B & \cdots & C B \end{bmatrix}$$

where $A, B$ and $C$ are the system state-space model matrices as indicated in Fig 1. Eq. (1e) represents the inequality constraints that apply to the system. Eq. (1f) and (1g) are moving endpoint constraints and impose the system to be at rest at the desired set point $y_{ref}$ within $N$ time steps.
The outcome of problem \( P_A(\hat{x}_t, N) \) is that it is either feasible or not. Infeasibility of \( P_A(\hat{x}_t, N) \) means that the system can not be at rest at the desired set point \( y_{ref} \) within \( N \) time steps while respecting system constraints. Therefore, an admissible set \( X(N) \) is defined:

\[
X(N) = \{ \hat{x}_t | P_A(\hat{x}_t, N) \text{is feasible} \} \tag{6}
\]

\( X(N) \) is the set of state systems from which \( y_{ref} \) can be reached within \( N \) time steps, while respecting all system constraints Eq. (1e).

'Problem B' denoted as \( P_B(\hat{x}_t, N) \), calculating the settling time \( N \), is defined as follows:

\[
V_B^*(\hat{x}_t, K^*) = \min N \quad \text{s.t.} \quad \hat{x}_t \in X(N) \tag{7a}
\]

\[
\max(N_{min}, K^*) \leq N \leq N_{max} \tag{7b}
\]

In Equation (7), \( N \) is bounded by \( N_{max} \) and the maximum of \( N_{min} \) and \( K^* \). To guarantee unconstrained solvability, \( N_{min} \) should be selected bigger than \( n/n_u \) with \( n \) the number of states and \( n_u \) the number of inputs [15]. Hence, at each time instant \( t \), 'problem B' minimizes \( N \) up to \( K^* \) if \( y_{ref} \) can be reached in \( K^* \) time steps (\( P_A(\hat{x}_t, K^*) \) is feasible) except if (i) \( y_{ref} \) can be reached in less than or equal to \( N_{min} \) time steps with \( K^* < N_{min} \), yielding \( N = N_{min} \) or if (ii) \( y_{ref} \) can’t be reached in \( K^* \) time steps, yielding \( N > K^* \).

The EOMPC optimization problem is a convex quadratic program (QP) which is solved using the qpOASES [4] open source on-line active set C++ software. In this paper, as we explained in the introduction, the blocking strategy [8] is utilized in EOMPC to deal with the conflicting requirements: (i) the number of decision variables has to be limited and (ii) a sufficiently large prediction time horizon has to be provided. However blocking yields a non-smooth optimal solution and as a result energy-optimality is achieved only approximately.

\[V_B^*(\hat{x}_t, K^*) = \min N \quad \text{s.t.} \quad \hat{x}_t \in X(N) \tag{7a}\]

\[\max(N_{min}, K^*) \leq N \leq N_{max} \tag{7b}\]

3 Pre-Stabilized EOMPC

The basic idea of the pre-stabilized EOMPC approach is illustrated on Fig. 2. Instead of solving the optimal control problem based on an open-loop system model, the pre-stabilized EOMPC approach calculates the optimal control sequence based on a closed-loop system model. Constructing the closed-loop system model is called pre-stabilization and it is the key to gain the computational advantage that the computational load is much less dependent on \( N_{max} \). The following paragraphs describe pre-stabilization in detail.

3.1 Pre-Stabilization

The idea of pre-stabilization comes from the literature on Generalized Predictive Control (GPC) [9]-[11], where it was introduced to modify the open-loop constrained optimal control problem to guarantee closed-loop stability. In [13] this idea was extended to state-space models and dead-beat state feedback and significant computational and numerical advantages were reported. In this paper, pre-stabilization in combined the EOMPC approach modifies the open-loop discrete-time system model to a closed-loop system model in order to achieve computational advantages similar as the ones reported in [13]. The concept of pre-stabilization is illustrated in the dashed block of Fig. 2. A dead-beat state-feedback controller [16] \( K \) is introduced such that the control signal \( u \) is defined as:

\[u_k = -K(x_k - x_{ref}) + r_k\]  \( \tag{8} \)

where \( x_{ref} \) is system state when the system is at rest at the desired set point \( y_{ref} \) and \( r_k \) is the new decision variable of the optimization problem. As a result of combing \( u_k \) with the open-loop discrete-time state-space system model shown in Fig. 1, the closed-loop discrete-time state-space system model is obtained and is shown in Eq. (9).

\[x_{k+1} = \Phi x_k + BK x_{ref} + Br_k\]  \( \tag{9a} \)

\[y_k = Cx_k\]  \( \tag{9b} \)

where \( \Phi = A - BK \).

According to the definition of deadbeat control [13], all poles of this closed-loop system model are located at zero and the following useful properties are obtained:

\[(A - BK)^n = \Phi^n = 0\]  \( \tag{10} \)

\[r_k = 0, \quad k = i, \ldots, N_{max} - 1\]

\[x_k = x_{ref}, \quad k = i + n, \ldots, N_{max}\]  \( \tag{11} \)

where \( n \) is the number of states and integer \( i \in \{1, \cdots, N_{max} - n\} \). As a result of property (2) Eq. (11), the two moving endpoint constraints Eq. (1f) and (1g) can be replaced by one endpoint constraint Eq. (12) to impose the system to be at rest at the desired set point \( y_{ref} \) within \( N \) time steps.
\[ r_k = 0, \quad k = N - n, \cdots, N_{\text{max}} - 1 \quad (12) \]

In the next section, we will explain how to achieve the computational advantage based on property (1) Eq. (10).

### 3.2 Two-Layer Optimization Problem

Similar to the EOMPC, the pre-stabilized EOMPC optimization problem is formulated as a two-layer optimization problem. The top layer is called 'Problem D' and determines the settling time \( N \). The feasibility problem of finding \( N \) is the second layer and is called 'Problem C'. The optimal control sequence is the last feasible solution of 'Problem C'.

'Problem C' denoted as \( P_C(\hat{x}_t, N) \) is defined in Eq. (13).

\[
\begin{align*}
V_C &= \min_{r} E_{\text{loss}}(u) \\
\text{s.t. } & x_0 = \hat{x}_t \\
& u = U_s x_0 + U_f x_{\text{ref}} + U_r r \\
& x = X_s x_0 + X_f x_{\text{ref}} + X_r r \\
& e_1 \leq H x + G u \leq e_2 \\
& r_k = 0, \quad k = N - n, \cdots, N_{\text{max}} - 1 \quad (13f)
\end{align*}
\]

In Equation (13), \( r = [r_0, \cdots, r_{N_{\text{max}} - 1}]^T \) is the decision variable over the prediction horizon \( N_{\text{max}} \) and is different from the system control signal \( u \). Based on the closed-loop discrete-time state-space system model Eq. (9), Eq. (13c) and (13d) define the system control sequence \( u = [u_0, \cdots, u_{N_{\text{max}} - 1}]^T \) and the state system sequence \( x = [x_0^T, \cdots, x_{N_{\text{max}}}]^T \) over the prediction horizon. They depend on the current state \( x_0 \), the steady state \( x_{\text{ref}} \) and the decision variable \( r \) with \( U_s, U_f, U_r, X_s, X_f \) and \( X_r \) defined in Eq. (14)-(19). Eq. (13e) specifies the system constraints. Eq. (13f) is a moving endpoint constraint and as illustrated in Eq. (11) it is sufficient to impose the system to be at rest at the desired set point within \( N \) time steps.

\[
U_s = -K \times \begin{bmatrix} I \\ \Phi \\ \Phi^2 \\ \vdots \\ \Phi^{N_{\text{max}} - 1} \end{bmatrix} \quad (14)
\]

\[
U_f = -K \times \begin{bmatrix} -I \\ \Phi^0 BK - I \\ (\Phi^1 + \Phi^0) BK - I \\ \vdots \\ (\Phi^{N_{\text{max}} - 1} + \cdots + \Phi^0) BK - I \end{bmatrix} \quad (15)
\]

\[
U_r = \begin{bmatrix} I & 0 & 0 & \cdots & 0 \\ -BK & I & 0 & \cdots & 0 \\ -B\Phi K & -BK & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -B\Phi^{N_{\text{max}} - 2} K & -B\Phi^{N_{\text{max}} - 3} K & -B\Phi^{N_{\text{max}} - 4} K & \cdots & I \end{bmatrix} \quad (16)
\]

\[
X_s = \begin{bmatrix} \Phi \\ \Phi^2 \\ \vdots \\ \Phi^{N_{\text{max}}} \end{bmatrix} \quad (17)
\]

\[
X_f = \begin{bmatrix} I \\ \Phi^1 + I \\ \Phi^2 + \Phi^1 + I \\ \vdots \\ (\Phi^{N_{\text{max}} - 1} + \cdots + I) \end{bmatrix} \times BK \quad (18)
\]

\[
X_r = \begin{bmatrix} B & 0 & \cdots & 0 \\ \Phi B & B & 0 & \cdots \\ \vdots & \vdots & \ddots & \vdots \\ \Phi^{N_{\text{max}} - 1} B & \Phi^{N_{\text{max}} - 2} B & \cdots & B \end{bmatrix} \quad (19)
\]

As we mentioned before, property (1) shown in Eq. (10) yields that \( \Phi^k = 0 \) for all \( k \geq n \). As a result, in matrices \( U_s, U_f, U_r, X_s, X_f \) and \( X_r \), all block elements proportional to \( \Phi^k (k \in n, \cdots, N_{\text{max}}) \) are equal to zero. This means that matrices \( U_s, U_r, X_s, X_f \) and \( X_r \) are sparse matrices.

Since 'Problem D' is defined in the same way as 'Problem B', it is not repeated here.

### 4 Numerical Validation

#### 4.1 Considered Test Setup

The considered system is the linear motor of the badminton robot setup developed by Flanders’ Mechatronics Technology Centre (FMTC) and is schematically represented in Fig. 3. This test setup is described in detail in [17] and [8]. This linear motor is used to position the 3 degrees of freedom badminton robot across the field and is the main energy consumer of this setup [8]. The other 2 axes of the robot are a rotational axis and a hit axes. The other main part of the setup is a stereo camera system used to detect the shuttle cock. Interception logic determines the reference position and corresponding motion time in order to hit back the shuttle. The motion time \( T \) is always smaller or equal to 1.5[s]. This limit is a direct result of the dimensions of the field and the operation of the camera system. The sampling time \( T_s = 10[ms] \). This means that the total prediction length \( N_{\text{max}} \) should be selected no less than \( T/T_s \) which is 150 in this case.
The dynamics of this linear motor relating the motor current and position are modelled as a double integrator, meaning that the linear motor friction is negligible. The discrete-time state-space model with sampling time $T_s = 10 \text{[ms]}$ is shown in Eq. (20).

\begin{align}
x_{k+1} &= Ax_k + Bu_k \quad (20a) \\
y_k &= Cx_k = x_k(2) \quad (20b) \\
v_k &= C_v x_k = x_k(1) \quad (20c)
\end{align}

in which $A = \begin{bmatrix} 1 & 0 \\ 0.01 & 1 \end{bmatrix}, B = \begin{bmatrix} 0.01 \\ 0.00005 \end{bmatrix}, C = \begin{bmatrix} 0 & 1 \end{bmatrix}, C_v = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and $k$ is the discrete time index. Input $u$ is acceleration $a[m/s^2]$, velocity $v[m/s]$ and position $y[m]$ are the first and the second state variables of the system respectively. Due to the limitations of the system, such as peak current limitations and the total length of the linear motor, the limits on the position, the velocity, and the acceleration, which are $\pm 1.9[m], \pm 3[m/s], \pm 30[m/s^2]$ respectively, are taken into account.

Because of the negligible friction, the copper losses determine the energy consumption, which is proportional to the square of the motor current, and hence proportional to the square of the system input $u$, yielding following energy consumption model:

\[ E_{\text{loss}}(u) = T_s \times c_e \times \sum_{k=0}^{N_{\text{max}}-1} u_k^2 \quad (21) \]

where $c_e = 1.57[W s^4/m^2]$ is a constant depending on the ohmic resistance of the motor windings.

### 4.2 Formulation of the Optimization Problem

First, the pre-stabilization of the open-loop discrete-time state-space model Eq. (20) is implemented. Using a pole placement approach [1], the deadbeat state-feedback controller $K = \begin{bmatrix} 150 & 1000 \end{bmatrix}$ is obtained. Thus the optimization variable $r$ defined in Eq. (8) and the closed-loop model defined in Eq. (9) are obtained.

**Problem C** of this time-constrained energy-optimal point-to-point motion is formulated equally to Eq. (13). In the object function Eq. (13a), $E_{\text{loss}}(u)$ defined in Eq. (21) is minimized. In Eq. (13c) and (13d), system control sequence $u = [u_0, \ldots, u_{N_{\text{max}}-1}]^T$ and system state sequence $x = [x_1^T, \ldots, x_{N_{\text{max}}}]^T$ over the prediction horizon are defined through Eq. (13c) and (13d) with:

\[
U_s = \begin{bmatrix} -150 & -10000 \\ 50 & 10000 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix}, \quad U_f = \begin{bmatrix} 150 & 10000 \\ -150 & -10000 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix}
\]

\[
U_r = \begin{bmatrix} 1 \\ -2 & 1 \\ 1 & -2 & 1 \\ 0 & \vdots \\ 0 & \vdots \\ 0 & \vdots \\ 0 & \vdots \\ 1 & -2 & 1 \end{bmatrix}
\]

\[
X_s = \begin{bmatrix} -0.5 & -1007 \\ 0.0025 & 0.5 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix}, \quad X_f = \begin{bmatrix} 1.5 & 1007 \\ 0.0075 & 0.5 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \end{bmatrix}
\]

\[
X_r = \begin{bmatrix} B \\ \Phi B & B \\ \Phi B & \Phi B \\ \Phi B & \vdots \\ \Phi B & B \end{bmatrix}
\]

with $B = \begin{bmatrix} 0.01 \\ 0.00005 \end{bmatrix}$, and $\Phi B = \begin{bmatrix} -0.01 \\ 0.00005 \end{bmatrix}$. Eq. (13e) specifies the system constraints on the acceleration, velocity, and position. As we know velocity $v$ and position $y$ are the first and the second state variables of the system, hence Eq. (13e) can be explicitly redefined as follows:

\[
-30 \leq u_k \leq 30, \quad k = 0, \ldots, N_{\text{max}} - 1
\]

\[
-3 \leq x_k \leq 3, \quad k = 1, \ldots, N_{\text{max}}
\]

In order to define the system constraints as a function of the decision variable $r$, Eq. (13c) and (13d) are substituted in Eq. (26) resulting in:
\[
\begin{bmatrix}
-30 \\
\vdots \\
-3 \\
-1.9 \\
\vdots 
\end{bmatrix} \leq U_s x_0 + U_f x_{ref} + U_r r \leq \begin{bmatrix}
30 \\
\vdots \\
3 \\
1.9 \\
\vdots 
\end{bmatrix} 
\tag{27a}
\]
\[
\begin{bmatrix}
-30 \\
\vdots \\
-3 \\
-1.9 \\
\vdots 
\end{bmatrix} \leq X_s x_0 + X_f x_{ref} + X_r r \leq \begin{bmatrix}
30 \\
\vdots \\
3 \\
1.9 \\
\vdots 
\end{bmatrix} 
\tag{27b}
\]

where \( x_0 \) and \( x_{ref} \) are known at each sampling instant. By defining \( V = [U_s x_0 + U_f x_{ref}, X_s x_0 + X_f x_{ref}] \), Eq. (27a) and (27b) are combined and reformulated as follows:

\[
\begin{bmatrix}
-30 \\
\vdots \\
-3 \\
-1.9 \\
\vdots 
\end{bmatrix} - V \leq \begin{bmatrix} U_r \end{bmatrix} \times r \leq \begin{bmatrix} 30 \\
\vdots \\
3 \\
1.9 \\
\vdots 
\end{bmatrix} - V \tag{28}
\]

which is equivalent to

\[
e_1 \leq F \times r \leq e_2 \tag{29}
\]

with

\[
F = \begin{bmatrix} U_r \\ X_r \end{bmatrix}, \quad e_1 = \begin{bmatrix} -30 \\
\vdots \\
-3 \\
-1.9 \\
\vdots 
\end{bmatrix} - V, \quad e_2 = \begin{bmatrix} 30 \\
\vdots \\
3 \\
1.9 \\
\vdots 
\end{bmatrix} - V
\]

Clearly matrices \( U_s, U_f, U_r, X_s, X_r \) and \( F \) are sparse matrices.

### 4.3 Simulation Results

In order to evaluate the pre-stabilized EOMPC, the following simulation experiment is considered. The robot has to move \( 1[m] \) within a required motion time \( T = 0.5[s] \), consuming as less energy as possible. The dashed line in Fig. 4(b) shows that the displacement of \( 1[m] \) is requested at \( t = 0.1[s] \). Fig. 4(a) shows the available motion time \( T \), which changes linearly from \( T = 0.5[s] \) to \( 0[s] \).

First the pre-stabilized EOMPC is compared to the EOMPC without applying the blocking. \( N_{max} = 150 \). Both implementations yield exactly the same motion, shown in Fig. 4(b), (c) and (d) (solid lines). The dashed lines in Fig. 4(c) and (d) indicate the imposed velocity and acceleration constraints respectively. The acceleration jumps to a maximum value and then linearly decreases to a minimal value at the end of the motion time. [18] shows that this acceleration profile corresponds to an energy-optimal motion for the considered system dynamics.

The CPU time shown in Fig. 4(e), is quite different for both implementations. As we mentioned before, at each sampling instant, the EOMPC optimal control problem, which is a convex QP, is solved using the active set strategy [19] qpOASES [20]. At each sampling instant, qpOASES solves one convex QP optimization problem. One of two functions 'init' and 'hotstart' can be selected depending on whether there is a change of the desired set point \( y_{ref} \) or not. If a new \( y_{ref} \) is requested, function 'init' is selected to solve the QP problem based on the current system state.

![Figure 4](image-url)

Figure 4. (a) Requested motion time\( T \), (b) position\( y \), (c) velocity\( v \), (d) acceleration\( a \) of the system and (e) CPU time.
and an initial guess of settling time $N$. Otherwise, function ‘hotstart’ is selected to solve the QP problem based on the feasible solution of previous QP problem as a hot start. This typically leads to the situation where the CPU time of calling function ‘init’ is considerably larger than calling function ‘hotstart’. At $t = 0.1[s]$ when the displacement of $1[m]$ is requested, the QP problem is solved by calling function ‘init’. The following QP problems are solved by calling function ‘hotstart’. However, in case the QP comprises dense constraints, the CPU time is mainly determined by the time requested to update the set of inequality constraints which are going to be active at the next iterate and hence large CPU times can occur. Since the pre-stabilization results in sparse constraints, the resulting CPU time has one peak (8.6[ms]) at $t = 0.1[s]$, and due to the hot starts, is significantly lower at the following time instants. EOMPC without pre-stabilization results in a QP with dense constraints, and hence the worst CPU time is 4 times larger than that of the pre-stabilized EOMPC which is 32.54[ms].

Remark that due to the large CPU times, EOMPC without pre-stabilization cannot be implemented in real-time at the considered sampling rate of 10[ms]. Applying the blocking strategy as described in [8] with $N_{max} = 40$ reduces the worst case CPU time below 10[ms]. This blocking however results in a non-smooth control input or acceleration signal, as shown in Fig. 5, and a 4% increase of the energy consumption: 157.13[J] for the EOMPC with blocking strategy and 150.78[J] for the pre-stabilized EOMPC approach.

5 Conclusion

This paper presents the pre-stabilized Energy-optimal MPC approach which is developed based on Energy-optimal MPC (EOMPC). These two approaches both aim at performing time-constrained energy-optimal point-to-point motions of linear time invariant (LTI) systems. Compared with EOMPC, the main advantage of the pre-stabilized EOMPC is that the computational load is much less dependent on the number of the decision variables of the optimization problem. This is achieved by modifying the open-loop discrete-time state-space system model employed in the formulation of MPC with a deadbeat state-feedback controller, hence resulting in a much sparser optimization problem. The numerical validation results show that both EOMPC and pre-stabilized EOMPC are capable of realizing time-constrained energy-optimal point-to-point motions. Using pre-stabilized EOMPC, smooth energy-optimal solutions are achieved and the computational efficiency is significantly improved.

Acknowledgements

This work has been carried out within the framework of projects IWT-SBO 80032 (LeCoPro) of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen). This work also benefits from K.U.Leuven-BOF PFV/10/002 Center-of-Excellence Optimization in Engineering (OPTEC), and the Belgian Programme on Interuniversity Attraction Poles, initiated by the Belgian Federal Science Policy Office (DYSCO), K.U.Leuven’s Concerted Research Action GOA/10/11.

References


