A hierarchical time-splitting approach for solving finite-time optimal control problems

Georgios Stathopoulos\textsuperscript{1}, Tamás Keviczky\textsuperscript{2} and Yang Wang\textsuperscript{3}

Abstract—We present a hierarchical computation approach for solving finite-time optimal control problems using operator splitting methods. The first split is performed over the time index and leads to as many subproblems as the length of the prediction horizon. Each subproblem is solved in parallel and further split into three by separating the objective from the equality and inequality constraints respectively, such that an analytic solution can be achieved for each subproblem. The proposed solution approach leads to a nested decomposition scheme, which is highly parallelizable. We present a numerical comparison with standard state-of-the-art solvers, and provide analytic solutions to several elements of the algorithm, which enhances its applicability in fast large-scale applications.

I. INTRODUCTION

Online optimization and optimal control methods are increasingly being considered for fast embedded applications, where efficient, reliable, and predictable computations involved in calculating the optimal solutions are a necessity. The potential use of optimal control in such embedded systems promises energy savings and more efficient resource usage, increased safety, and improved fault detection. The range of application areas that can benefit from embedded optimization include the mechatronics, automotive, process control and aerospace sectors \cite{1}. The promise of unprecedented performance and capabilities in these applications, which typically rely on large-volume, real-time embedded control systems, has fueled recent research efforts towards fast and parallel optimization solvers.

One of the main research directions aim at developing special-purpose optimization solvers that target typical control or estimation problems arising in optimal control. Parallel solutions to systems of linear equations appearing in interior-point, and active set methods have been studied in \cite{2}–\cite{5}. In this work we consider a quadratic finite-time optimal control problem for discrete-time systems with constrained linear dynamics, which appears in typical model predictive control problems \cite{6}. We investigate and develop different parallelizable algorithms using operator splitting techniques \cite{7}, \cite{8} that have recently shown great promise for speeding up calculations involved in computing optimal solutions with medium accuracy \cite{9}–\cite{11}. Our approach relies on a hierarchical splitting up of the specially structured finite-time optimal control problem. The first split is performed over the time index and leads to as many subproblems as the length of the prediction horizon. Each subproblem can then be solved in parallel and further split into three by separating the objective from the equality and inequality constraints respectively, such that an analytic solution can be achieved for each subproblem. The proposed solution approach leads to a nested decomposition scheme, which is highly parallelizable. The proposed three-set splitting method does not only solve the particular quadratic programs (QPs) that appear in the update steps of the time-splitting algorithm efficiently, but also provides a compact, standalone alternative for solving generic QPs.

The paper is structured as follows. Section \textbf{II} presents the main idea behind the time-splitting optimal control approach for parallel computations, using the Alternating Direction Method of Multipliers and deriving the exact formulas required for each subproblem and update step. In Section \textbf{III} we propose an alternative scheme for solving the QPs with general polyhedral constraints that arise in the time-splitting update steps (or for any other generic QP). The two splitting schemes are combined in a hierarchical fashion in Section \textbf{IV} and numerical experiments are performed in Section \textbf{V} to compare its performance with some advanced solvers in the literature. Section \textbf{VI} concludes the paper.

II. TIME-SPLITTING OPTIMAL CONTROL

A. Problem Formulation

We consider the following finite-time optimal control problem formulation that arises in typical model predictive control applications:

\begin{align}
\text{minimize} & \quad \frac{1}{2} \sum_{t=0}^{N} \left( x_t^T Q x_t + u_t^T R u_t \right) \\
\text{subject to} & \quad (x_t, u_t) \in \mathcal{X}_t \times \mathcal{U}_t, \quad t = 0, \ldots, N \\
& \quad x_{t+1} = Ax_t + Bu_t + ct, t = 0, \ldots, N - 1
\end{align}

where the decision variables are the states $x_t \in \mathbb{R}^n$, and the inputs $u_t \in \mathbb{R}^m$ of the system for $t = 0, \ldots, N$. The index $t$ denotes time, and the system evolves according to linear dynamics constraint (1c) where $c_t \in \mathbb{R}^n$ is considered to be a known disturbance. Here, $N$ is the prediction horizon and $Q \in \mathbb{R}^{n \times n}, R \in \mathbb{R}^{m \times m}$ are symmetric matrices. The stage cost functions in (1a) are convex quadratic with $Q \succeq 0$ and $R \succ 0$. The stage-wise state-input pairs are constrained to reside within polyhedra (1b) denoted by $\mathcal{X}_t$ and $\mathcal{U}_t$, respectively. These are constraint sets defined by linear inequalities that involve states and inputs at the same sample time index.

---

\textsuperscript{1}G. Stathopoulos is with Laboratoire d’Automatique, EPFL, CH-1015 Lausanne, Switzerland, georgios.stathopoulos@epfl.ch
\textsuperscript{2}T. Keviczky is with the Delft Center for Systems and Control, Delft University of Technology, Delft, CD 2628, The Netherlands, t.keviczky@tudelft.nl
\textsuperscript{3}Y. Wang is with Stanford University, Stanford, CA 94305, USA, yang1024@gmail.com
Motivated by the principles of operator splitting methods (see [12] for details and relevant references), we propose to split the problem (1) into $N+1$ smaller stage-wise subproblems that can be solved in parallel. This requires breaking the coupling that appears due to the dynamics. We introduce a copy of each variable that couples the dynamics equations in order to allow such a splitting into subproblems, and subsequently impose a consensus constraint on the associated complicating variables and their copies. This leads to the following equivalent formulation of (1), where the complicating variables that are used to perform the splitting are clearly highlighted:

$$\begin{align*}
\text{minimize} \quad & \frac{1}{2} \sum_{t=0}^{N} \left( x_{t}^{(i)}^T Q_{t} x_{t}^{(i)} + u_{t}^{(i)}^T R_{t} u_{t}^{(i)} \right) \\
\text{subject to} \quad & (x_{t}^{(i)}, u_{t}^{(i)}) \in \mathcal{X}_{t} \times \mathcal{U}_{t}, \quad t = 0, \ldots, N \quad (2a) \\
& x_{t+1} = A_{t} x_{t}^{(i)} + B_{t} u_{t}^{(i)} + c_{t}, \quad t = 0, \ldots, N-1 \quad (2b) \\
& x_{0}^{(i)} = x_{\text{init}}, \quad \ddot{x}_{t+1} = x_{t}^{(i)}, \quad t = 0, \ldots, N-1, \quad (2c)
\end{align*}$$

where the subscript $t$ of the decision variables $x_{t}^{(i)}, u_{t}^{(i)}$ indicates the time index and the superscript $(i)$ denotes the group or subproblem where the variable belongs to. Hence, each subproblem contains three variables, the current state and input as well as a prediction of the state for the next time instant. The introduced complicating variable $\ddot{x}$ acts as a ‘global’ variable that brings the local copies $x_{t}^{(i-1)}$ and $x_{t}^{(i)}$ in agreement, i.e., $\ddot{x}_{t} = x_{t}^{(i-1)} = x_{t}^{(i)}$. The time-splitting idea is graphically depicted in Figure 1.

**Fig. 1.** The idea of the time splitting algorithm. Whenever the dynamics introduce a coupling of variables, they are decoupled via a slack variable $\ddot{x}_{t} = x_{t}^{(i-1)}$.

**B. The time-splitting algorithm**

In order to use a more compact formulation, we will denote the decision variables in (2) corresponding to each subproblem using

$$\ddot{x}_{t} = (x_{t}^{(i)}, u_{t}^{(i)}, x_{t+1}^{(i)}), \quad (3)$$

where $\ddot{x}_{t} \in \mathbb{R}^{2n+m}$. We also introduce dual variables to deal with the consensus equality constraints:

- $\tilde{w}_{t}$ associated with $x_{t}^{(i)} = \ddot{x}_{t}$, $t = 1, \ldots, N$
- $\tilde{v}_{t}$ associated with $x_{t}^{(i-1)} = \ddot{x}_{t}$, $t = 1, \ldots, N$.

In order to rewrite the finite-time optimal control problem in a more compact form, we define the following matrices:

$$\begin{align*}
P_{t} &= \begin{bmatrix} Q_{t} & 0 & 0 \\ 0 & R_{t} & 0 \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{(2n+m) \times (2n+m)}, \quad (4a) \\
F_{t} &= \begin{bmatrix} -A_{t} & -B_{t} & I \end{bmatrix} \in \mathbb{R}^{n \times (2n+m)}, \quad (4b) \\
G_{0} &= \begin{bmatrix} I & 0 & 0 \end{bmatrix} \in \mathbb{R}^{n \times (2n+m)}, \quad (4c) \\
G_{t} &= \begin{bmatrix} 0 & 0 & I \end{bmatrix} \in \mathbb{R}^{n \times (2n+m)}. \quad (4d)
\end{align*}$$

We use the ALternating Direction Method of Multipliers (ADMM) [13], [14] in order to arrive at a solution approach that is amenable to parallel implementation. The updates involved in the ADMM algorithm include forming the augmented Lagrangian of the problem and minimizing over the primal variables $\ddot{x}_{t}, t = 0, \ldots, N$ and $\ddot{z}_{t}, t = 1, \ldots, N$, followed by updating the dual variables $\tilde{w}_{t}$ and $\tilde{v}_{t}$, $t = 1, \ldots, N$. The three main steps of the algorithm are performed in an iterative fashion and are described next in detail. We use $k$ to denote the algorithm’s loop counter. The termination criterion based on primal and dual tolerances are provided and the analytic derivation of the formulas in each step is presented in the Appendix.

**Step 1: Solving $N+1$ QP subproblems for the primal variables in $\ddot{x}_{t}$**

Minimization of the augmented Lagrangian over the primal variables $\ddot{x}_{t}$ results in $N+1$ stage-wise quadratic programs (QPs):

- For the subproblem associated with the time instant $t = 0$, we need to solve

$$\begin{align*}
\text{minimize} \quad & \frac{1}{2} \ddot{x}_{0}^T P_{0} \ddot{x}_{0} - \rho \ddot{x}_{0}^T (G_{0} \ddot{x}_{0} - z_{1}^{(1)}) \\
& + (\rho/2) \|G_{0} \ddot{x}_{0} - z_{1}^{(1)}\|_{2}^{2} \\
\text{subject to} \quad & \ddot{x}_{0} \in \mathcal{F}_{0} \\
& G_{0} \ddot{x}_{0} = x_{\text{init}} \\
& F_{0} \ddot{x}_{0} = c_{0} \quad (5)
\end{align*}$$

with variable $\ddot{x}_{0}$.

- Similarly, we need to solve the following QPs for all the other groups of variables $\ddot{x}_{t}, t = 1, \ldots, N-1$:

$$\begin{align*}
\text{minimize} \quad & \frac{1}{2} \ddot{x}_{t+1}^T P_{t+1} \ddot{x}_{t+1} - \rho \ddot{x}_{t+1}^T (G_{t+1} \ddot{x}_{t+1} - z_{t+1}^{(t+1)}) \\
& + (\rho/2) \|G_{t+1} \ddot{x}_{t+1} - z_{t+1}^{(t+1)}\|_{2}^{2} \\
& - \rho \ddot{x}_{t+1}^T (G_{t+1} \ddot{x}_{t+1} - z_{t+1}^{(t+1)}) \\
& + (\rho/2) \|G_{t+1} \ddot{x}_{t+1} - z_{t+1}^{(t+1)}\|_{2}^{2} \\
\text{subject to} \quad & \ddot{x}_{t} \in \mathcal{F}_{t} \\
& F_{t} \ddot{x}_{t} = c_{t} \quad (6)
\end{align*}$$

$$\begin{align*}
\ddot{z}_{1} = x_{t}^{(i-1)} = z_{t}, \quad t = 1, \ldots, N.
\end{align*}$$
• For the subproblem associated with the final time instant \( t = N \), we need to solve

\[
\begin{align*}
\text{minimize} & \quad (1/2)\tilde{x}_N^T P_N \tilde{x}_N - \rho \tilde{w}_N^T (G_0 \tilde{x}_N - \tilde{z}_N) \\
\text{subject to} & \quad \tilde{x}_N \in \mathcal{C}_N
\end{align*}
\]

with variable \( \tilde{x}_N \).

The polyhedral sets \( \mathcal{C}_t, t = 0, \ldots, N \) are defined as

\[
\mathcal{C}_t = \mathcal{X}_t \times \hat{\mathcal{Y}}_t \times \hat{\mathcal{Y}}_{t+1} \subseteq \mathbb{R}^{2n+m}
\]

and the variable \( \rho > 0 \) is a parameter of the algorithm.

Remark 1: Notice that for the time instant \( N \) the decision variables of the QP actually simplify to \( \tilde{x}_N = x_N \) and \( \mathcal{C}_N = \mathcal{X}_N \), but we keep the same notation for simplicity (and without loss of generality).

Step 2: Averaging

The update of the ‘global’ primal variables \( \tilde{z}_t, t = 1, \ldots, N \) is derived from a simple quadratic minimization problem, the solution of which turns out to be an average of the predicted \( (x_t^{(t-1)}) \) and current \( (x_t^{(t)}) \) state

\[
\tilde{z}_{k+1}^t = \frac{G_0 x_{k+1}^t + G_1 x_{k}^t}{2}, \quad t = 1, \ldots, N.
\]

This intuitively makes sense, since the global variable can be obtained by collecting the local (primal) ones and computing the best estimate based on their values.

Step 3: Dual update

The dual updates can be expressed as

\[
\begin{align*}
\tilde{w}_t^{k+1} &= \tilde{w}_t^k - G_0 \tilde{v}_t^{k+1} + \tilde{z}_t^{k+1}, \\
\tilde{v}_t^{k+1} &= \tilde{v}_t^k - G_1 \tilde{v}_t^{k+1} + \tilde{z}_t^{k+1},
\end{align*}
\]

and we defined the vectors

\[
\begin{align*}
\mathbf{x}_{\text{pri}} &= \begin{bmatrix} \bar{x}_0 \\ \bar{x}_1 \\ \vdots \\ \bar{x}_N \end{bmatrix}, \quad \mathbf{z}_{\text{pri}} = \begin{bmatrix} \bar{z}_1 \\ \bar{z}_2 \\ \vdots \\ \bar{z}_N \end{bmatrix}, \quad \mathbf{v}_{\text{dual}} = \begin{bmatrix} \bar{v}_1 \\ \bar{v}_2 \\ \vdots \\ \bar{v}_N \end{bmatrix}
\end{align*}
\]

The residual matrices \( A_{\text{res}} \) and \( B_{\text{res}} \) and the residual vector \( c_{\text{res}} \) are

\[
\begin{align*}
A_{\text{res}} &= \begin{bmatrix} G_1 & 0 & \cdots & 0 \\
0 & G_0 & \cdots & 0 \\
0 & 0 & \ddots & \vdots \\
0 & 0 & \cdots & G_0 \\
0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{T2n \times (N+1)(2n+m)}, \\
B_{\text{res}} &= \begin{bmatrix} -I & 0 & \cdots & 0 \\
0 & -I & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & -I \\
0 & 0 & \cdots & -I \end{bmatrix} \in \mathbb{R}^{N2n \times Nn},
\end{align*}
\]

and \( c_{\text{res}} \in \mathbb{R}^{N2n} \) is zero.

The three update steps described above are fully parallelizable at each iteration \( k \). Assuming \( N+1 \) processors are available, then processor \( \Pi_i \) would need to execute the following actions for \( t = 0, \ldots, N-1 \):

1) Receive the estimate \( \tilde{z}_{t+1}^{k+1} \) and from neighboring processor \( \Pi_{t+1} \).
2) Compute \( \tilde{x}_{t+1}^{k+1} \).
3) Receive the estimate \( \tilde{w}_{t+1}^{k+1} \) from neighboring processor \( \Pi_{t} \) and compute \( \tilde{z}_{t+1}^{k+1} \).
4) Compute \( \tilde{v}_{t+1}^{k+1} \) and \( \tilde{v}_{t+1}^{k+1} \).
5) Communicate \( \tilde{x}_{t+1}^{k+1} \) to processors \( \Pi_{t-1} \) and \( \Pi_{t+1} \), and \( \tilde{z}_{t+1}^{k+1}, \tilde{v}_{t+1}^{k+1} \) to processor \( \Pi_{t-1} \).

The above scheme suggests that each processor \( \Pi_i, t = 1, \ldots, N-1 \) interacts with the two neighboring processors \( \Pi_{t-1} \) and \( \Pi_{t+1} \). Processors \( \Pi_0 \) and \( \Pi_N \) communicate only with processors \( \Pi_1 \) and \( \Pi_{N-1} \), respectively. After updating all variables, a gather operation follows in order to compute the residuals and check the termination criterion.

III. THREE-SET SPLITTING QP SOLVER

A. Motivation

The time-splitting algorithm presented in the previous section decomposes the centralized finite-time optimal control problem so that it can be solved using multiple parallel processors. However, the updates for the primal variables...
\[ \dot{x}_t, t = 0, \ldots, N \] given in (5), (6) and (7) involve solving a QP at each iteration of the algorithm. Even though several fast interior point solvers exist for this purpose (see e.g., [15]), these are mostly suitable for only a limited number of variables. Although recently more computationally efficient schemes that scale better with the problem size have been developed, they are restricted to cases where simple box constraints are considered [8], [10], [16]. In order to achieve fast computations in an embedded control environment, other generic solution methods would be preferred.

In this section we propose an alternative scheme for solving the QPs with general polyhedral constraints that arise in the previous section. We propose to perform yet another type of splitting approach, which splits the state-input variables of the QP in three sets. One set involves the variables that appear in the objective function, another set includes those that appear in the dynamics equality constraints, and the last set contains variables from the inequality constraints. In this way, we solve three simpler subproblems instead of the single general QP. Since several variables are shared among the subproblems, their solutions must be in consensus again to ensure consistency.

An important element of the proposed method is the introduction of an extra slack variable, which allows to get an analytic solution for the subproblem associated with the inequality constraints. Using this variable, the projection on any polyhedral set can be practically rewritten as a projection onto the nonnegative orthant. Besides this feature, the proposed splitting exploits structure in the resulting matrices and thus modern numerical linear algebra methods can be employed for speeding up the computations.

### B. Problem setup

We consider a QP of the form

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2}x^T M x + q^T x + r \\
\text{subject to} & \quad A x = b \\
& \quad H x \preceq h,
\end{align*}
\]

with decision variable \( x \in \mathbb{R}^n \), where \( M \in \mathbb{S}^n_+ \), \( q \in \mathbb{R}^n \), \( r \in \mathbb{R} \), \( A \in \mathbb{R}^{m \times n} \), \( b \in \mathbb{R}^m \), \( H \in \mathbb{R}^{p \times n} \), \( h \in \mathbb{R}^p \) and \( \mathbb{S}^n_+ \) is the cone of positive semidefinite matrices of dimension \( n \).

In order to apply a variable splitting idea for this problem we first replicate all variables appearing in \( (14) \) three times, introducing three different sets for which we must ensure consensus. Furthermore, we use a slack variable to remove the polyhedral constraint and transform it into a projection operation onto the nonnegative orthant. We define the following sets of variables:

- **First set - objective:** \( x^I \)
- **Second set - equality constraints:** \( x^{II} \)
- **Third set - inequality constraints:** \( x^{III} \)

- **First global variable:** \( z = x^I = x^{II} = x^{III} \)
- **Second global variable:** \( y = h - H x^{III} \)
- **First set of dual variables:** \( \tilde{z}^I, \tilde{z}^{II}, \tilde{z}^{III} \) associated with \( z = x^I = x^{II} = x^{III} \), respectively.
- **Second set of dual variables:** \( \tilde{y} \) associated with \( y = h - H x^{III} \)

Using the above sets of variables problem (14) can be restated in the equivalent form

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2}x^T M x + q^T x + r \\
\text{subject to} & \quad A x = b \\
& \quad y = h - H x^{III}, \quad y \succeq 0 \\
& \quad x^I = x^{II} = x^{III} = z,
\end{align*}
\]

with variables \( x^I, x^{II}, x^{III}, z \in \mathbb{R}^n, y \in \mathbb{R}^p \). The dual variables are of dimensions \( \tilde{z} \in \mathbb{R}^n, \tilde{y} \in \mathbb{R}^p \).

### C. The proposed three-set splitting algorithm

The proposed algorithm consists of iterative updates to the three ADMM steps similarly to the case of the time-splitting approach in Section II-B, namely one for the (local) primal variables \( x^I, x^{II}, x^{III} \), one for the (global) primal variables \( z \) and \( y \) and one for the dual variables \( \tilde{z}^I, \tilde{z}^{II}, \tilde{z}^{III} \) and \( \tilde{y} \).

We provide the algorithm’s steps below along with some clarifying comments. The analytic derivations are presented in the Appendix.

**Step 1:** Solving three subproblems for the primal variables \( x^I, x^{II}, x^{III} \)

In all three cases we have to solve simple, unconstrained QPs. The updates are:

\[
(16) \quad (x^I)^{k+1} = (M + \rho I)^{-1}(\rho(z^k + (\tilde{z}^I)^k) - q)
\]

\[
(17) \quad \begin{bmatrix}
\rho I & A^T \\
A & 0
\end{bmatrix} \begin{bmatrix}
(x^{II})^{k+1} \\
\nu
\end{bmatrix} = \begin{bmatrix}
\rho(z^k + (\tilde{z}^{II})^k) \\
-b
\end{bmatrix},
\]

where \( \nu \) is the dual variable associated with the equality constraint \( A x^{II} = b \), and

\[
(18) \quad (x^{III})^{k+1} = (H^T H + I)^{-1}(H^T (h - y^k - y^k) + \\
\tilde{z}^I + (\tilde{z}^{III})^k).
\]

The matrices \( M + \rho I \) and \( H^T H + I \) are symmetric positive definite due to the regularization terms. This means that, instead of directly inverting the matrices, we can save computational effort by taking the Cholesky factorization, i.e., write the matrix as a product of a lower triangular matrix and its transpose (see, e.g., [17]). Furthermore, the matrix

\[
\begin{bmatrix}
\rho I & A^T \\
A & 0
\end{bmatrix}
\]

is a KKT matrix and \( \rho I > 0 \). Hence we can exploit its structure and use block elimination to solve the KKT system (see [18, App. C]).
be pre-factorized and then used in every solve step. The right-hand sides are the only parts that change in the update loop.

**Step 2: Averaging and projection**

The update for $z$ is

$$
\tilde{z}^{k+1} = \frac{1}{3} \sum_{i=1}^{III} (x_i)^{k+1},
$$

while for $y$ it is

$$
y^{k+1} = \left(h - H \left(x_{III}^{k+1} - y^k\right)\right) + .
$$

The $z$-update is an averaging over the three sets of the primal variables $x_I, x_{II}, x_{III}$, while the $y$-update is the solution of a proximal minimization problem (see Appendix), resulting in a projection onto the nonnegative orthant, denoted by $(\cdot)_+$.  

**Step 3: Dual update**

The update for the dual variables $\tilde{y}_i$ is

$$
\tilde{z}^{k+1} = (\tilde{z}^k - (x_i)^{k+1} + \epsilon^{k+1}_i), \quad i = I, II, III.
$$

Similarly, the update for $\tilde{y}$ is

$$
\tilde{y}^{k+1} = \tilde{y}^k + y^{k+1} - h + H \left(x_{III}^{k+1}\right).
$$

**Termination criterion**

The primal and dual tolerances $\epsilon^{pri}$ and $\epsilon^{dual}$ are given by

$$
\begin{align*}
\epsilon^{pri} &= \epsilon^{abs} \sqrt{3n + p} + \epsilon^{rel} \max \left(\|A_{res}(x^I, x^{II}, x^{III})\|_2, \|B_{res}(z, y)\|_2, \|c_{res}\|_2\right), \\
\epsilon^{dual} &= \epsilon^{abs} \sqrt{3n + p} + \epsilon^{rel} \max \left(\|A_{res}(y, \tilde{z}^I, \tilde{z}^{II}, \tilde{z}^{III})\|_2\right).
\end{align*}
$$

The residual matrices $A_{res}$ and $B_{res}$ and the residual vector $c_{res}$ are

$$
A_{res} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}, \quad B_{res} = \begin{bmatrix} -I & 0 \\ -I & 0 \\ 0 & I \end{bmatrix},
$$

and

$$
c_{res} = (0, 0, 0, h),
$$

where $A_{res} \in \mathbb{R}^{(3n+p) \times 3n}$, $B_{res} \in \mathbb{R}^{(3n+p) \times (n+p)}$ and $c_{res} \in \mathbb{R}^{3n+p}$.

**Remark 2:** For the QP corresponding to the last sample time $t = N$ (7), the algorithm simplifies to splitting into two sets (objective and inequality constraints), since there are no dynamics equality constraints. The updates and residuals follow directly from the more generic case presented above.

**IV. HIERARCHICAL TIME-SPLITTING OPTIMAL CONTROL**

It is a natural idea to combine the two splitting algorithms (time-splitting and three-set splitting) that were introduced in the preceding two sections in order to speed up the solution of the finite-time optimal control problem (1). This can be accomplished via a nested decomposition scheme, where we employ the *three-set splitting* algorithm to solve the QPs (5), (6) and (7) appearing in Step 1 of the *time-splitting* algorithm. The idea is graphically depicted in Figure 2.

If we rewrite the generalized inequality constraints appearing in the problem formulation as

$$
(x_t(t), u_t(t)) \in \mathcal{E}_t \Leftrightarrow H_t \tilde{x}_t \leq h_t, \quad t = 0, \ldots, N,
$$

then the QPs can be written in the form described by (14), where we consider the following relations:

- For $\tilde{x}_0$, Eq. (5):

  $$
  M = P_0 + \rho G_0^T G_1, \quad q = -\rho G_0^T (\tilde{z}_1 + \tilde{v}_1), \quad r = (\rho/2)\tilde{z}_1^T \tilde{z}_1 + \rho \tilde{v}_1^T \tilde{z}_1, \quad A = (G_0, F_0), \quad b = (x_{init}, c_0), \quad H = \begin{bmatrix} H_0 & 0 \end{bmatrix}, \quad h = h_0.
  $$

- For $\tilde{x}_t, t = 1, \ldots, N-1$, Eq. (6):

  $$
  M = P_t + \rho (G_0^T G_0 + G_1^T G_1), \quad q = -\rho (G_0^T (\tilde{z}_t + \tilde{w}_t) + G_1^T (\tilde{z}_{t+1} + \tilde{v}_{t+1})), \quad r = (\rho/2)(\tilde{z}_t^T \tilde{z}_t + \tilde{v}_t^T \tilde{z}_t) + \rho (\tilde{w}_t^T \tilde{z}_{t+1} + \tilde{v}_t^T \tilde{z}_t), \quad A = F_t, \quad b = c_t, \quad H = \begin{bmatrix} H_t & 0 \end{bmatrix}, \quad h = h_t.
  $$

- For $\tilde{x}_N$, Eq. (7):

  $$
  M = P_N + \rho G_0^T G_0, \quad q = -\rho G_0^T (\tilde{z}_N + \tilde{w}_N), \quad r = (\rho/2)\tilde{z}_N^T \tilde{z}_N + \rho \tilde{w}_N^T \tilde{z}_N, \quad A = 0, \quad b = 0, \quad H = \begin{bmatrix} H_N & 0 \end{bmatrix}, \quad h = h_N.
  $$

For each iteration of the time-splitting algorithm, the three-set splitting algorithm runs in an inner loop until it converges. The quality of this convergence, i.e., the choice of the primal and dual tolerances of the inner loop (23) will affect the quality of the global solution.

A method that enables substantial speedup of the algorithm is warm starting. Since the three-set splitting algorithm will run for every iteration of the time-splitting algorithm, we can warm-start each QP with its previous solution. In this way, we can achieve a significant reduction in the number of iterations needed for the convergence of the inner loop.
V. NUMERICAL RESULTS

We consider three, randomly generated, numerical examples to illustrate the performance of the algorithm. The examples vary in terms of the number of decision variables involved. The systems considered are linear and time-invariant. We impose constraints on the difference between two consecutive states at each time instant of the form

\[ x_{t,i} - x_{t,i-1} \leq dx, \]

where \( x_{t,i} \in \mathbb{R}, i = 1, \ldots, n, t = 0, \ldots, N \) and box constraints on the inputs, i.e.,

\[ \|u_t\|_\infty \leq u_{max}, t = 0, \ldots, N - 1. \]

By adjusting the level of the disturbance \( c_t \) in several time instances, we ensure activation of the constraints along the horizon.

For the simulations, we used an Intel Core i7 processor running at 1.7 GHz. We compared a C-implementation of our algorithm with using CVX [19], a parser-solver that uses SDPT3 [20]. For our method, the tolerances for both the outer and inner algorithms are set as \( \varepsilon^{pr} = 10^{-4} \) and \( \varepsilon^{dual} = 10^{-3} \). The parameter \( \rho \) was set after some simple tuning. The linear systems appearing in (16), (17) and (18) were solved by first factorizing the matrices off-line, using Tim Davis’s sparse package [21]–[23] (see also [8]). The finite-time optimal control problem was solved only once and all the primal and dual variables were initialized at zero. However, the inner algorithm was warm-started at every iteration of the outer algorithm to the values acquired from the previous iteration. No relaxation or any other variance of the iterations was used. The numerical results are summarized in Table I, where the computation times are reported in ms.

We can observe that, in the case of the small system, even when solving the problem on a single thread, the computation times are smaller than those of CVX. As the problem scales, the computations have to be parallelized in order to gain a significant advantage. More specifically, we expect the following speedup factors: 13 and 31 times faster in the case of the small problem (for the corresponding tolerances set to \( 10^{-4} \) and \( 10^{-3} \) respectively). For the medium-sized problem the speedups are by a factor of 10.5 and 24.6, and a factor of 5.4 and 11.7 for the large-scale problem, respectively.

In addition, we could observe that the factorization times are negligible in all cases, since the matrices being factorized are not large. Concerning the three-set splitting algorithm, only the average computational times are indicated over all iterations required to solve the problem.
VI. CONCLUSIONS

In this paper, we proposed an algorithm that solves a centralized convex finite-time optimal control problem making use of operator splitting methods, and, more specifically, the Alternating Direction Method of Multipliers. The initial problem is split into as many subproblems as the horizon length, which then can be solved in parallel.

The resulting algorithm is composed of three steps, including one where several QPs have to be solved. In this respect, we proposed another method, based again on operator splitting, that is applicable to QPs of any size, involving polyhedral constraints. This algorithm exploits the structure of the problem, leading to fast solutions.

The combination of the proposed algorithms results in a nested decomposition scheme for solving the aforementioned finite-time optimal control problems over several parallel processors.

Our numerical experiments suggest that the proposed hierarchical decomposition approach provides significant speed-up in computational time required for medium accuracy solutions for the class of problems considered. In our future work we intend to perform an even more extensive comparison with very recent tailor-made computational tools, and implement the algorithm on a parallel computing platform to obtain more accurate and representative computational time measurements.

APPENDIX

Derivation of the time-splitting algorithm updates

We solve the relaxed version of the convex optimization problem (2) by formulating the augmented Lagrangian with respect to the additional equality constraints, using the matrices and vectors defined in (3) and (4). The augmented Lagrangian can be written as

\[ L_\rho(\tilde{x}_t, x_{\text{init}}, \tilde{w}_t, \tilde{y}_t, \tilde{z}_t, \nu_t) := \]

\[ \sum_{t=0}^{N} \left( \frac{1}{2} \tilde{x}_t^T P \tilde{x}_t + I_{\tilde{x}_t}(\tilde{x}_t) \right) + \nu_t^T (\tilde{x}_0 - x_{\text{init}}) \]

\[ + \sum_{t=0}^{N-1} \nu_{t+1}^T (F \tilde{x}_t - c_t) \]

\[ + \sum_{t=0}^{N-1} \left( -\rho \tilde{v}_{t+1}^T (G_1 \tilde{x}_t - \tilde{z}_{t+1}) + \frac{\rho}{2} \| G_1 \tilde{x}_t - \tilde{z}_{t+1} \|^2_2 \right) \]

\[ + \sum_{t=1}^{N} \left( -\rho \tilde{w}_t^T (G_0 \tilde{x}_t - \tilde{z}_t) + \frac{\rho}{2} \| G_0 \tilde{x}_t - \tilde{z}_t \|^2_2 \right). \]

The dual variables \( \nu_t \in \mathbb{R}^n, t = 0, \ldots, N \) are associated with the equality constraints. We define the indicator function for the polyhedral set \( \mathcal{C}_t \subset \mathbb{R}^{2n+m} \) as

\[ I_{\mathcal{C}_t}(\tilde{x}_t) = \begin{cases} 0 & \tilde{x}_t \in \mathcal{C}_t \\ \infty & \text{otherwise} \end{cases} \]

The augmented Lagrangian is minimized (and maximized) over the primal (and dual) variables in an iterative manner, for each iteration of the algorithm.

By treating the dynamics and inequality constraints as explicit and minimizing with respect to \( \tilde{x}_t, t = 0, \ldots, N \), we end up with the stage-wise QPs (5), (6), (7).

The update of \( \tilde{z}_t \) is given by solving

\[ z_t^{k+1} = \arg\min_{z_t} \left( -\rho \tilde{v}_t^T (G_1 z_t^{k+1} - \tilde{z}_t) \right. \]

\[ + \frac{\rho}{2} \| G_1 z_t^{k+1} - \tilde{z}_t \|^2_2 \]

\[ - \rho \tilde{v}_t^T (G_0 z_t^{k+1} - \tilde{z}_t) + \frac{\rho}{2} \| G_0 z_t^{k+1} - \tilde{z}_t \|^2_2 \] \]

\[ = \arg\min_{z_t} \left( \frac{\rho}{2} \| z_t - G_1 z_t^{k+1} + \tilde{v}_t \|^2_2 + \frac{\rho}{2} \| z_t - G_0 z_t^{k+1} + \tilde{v}_t \|^2_2 \right) \]

\[ = \frac{G_0 \tilde{x}_t^{k+1} + G_1 z_t^{k+1} - \tilde{v}_t - \tilde{w}_t^k}{2}, \quad t = 1, \ldots, N \] (26)

Rearranging the terms in (25) and completing the squares as before, the dual updates for \( \tilde{w}_t \) and \( \tilde{v}_t \) result by solving

\[ \tilde{w}_t^{k+1} = \arg\max_{\tilde{w}_t} \left( \frac{\rho}{2} \| \tilde{w}_t - G_0 \tilde{x}_t^{k+1} + z_t^{k+1} \|^2_2 - \frac{\rho}{2} \| \tilde{w}_t \|^2_2 \right), \]

\[ \tilde{v}_t^{k+1} = \arg\max_{\tilde{v}_t} \left( \frac{\rho}{2} \| \tilde{v}_t - G_1 z_t^{k+1} + z_t^{k+1} \|^2_2 - \frac{\rho}{2} \| \tilde{v}_t \|^2_2 \right) \]

| Size | small | medium | large |
|------|-------|--------|-------|
| states \( n \) | 10 | 20 | 50 |
| inputs \( m \) | 10 | 10 | 40 |
| horizon length \( N \) | 10 | 30 | 60 |
| total variables | 220 | 900 | 5400 |
| active box constraints | 15 | 25 | 50 |
| active inequality constraints | 5 | 6 | 20 |
| CVX solve time | 2430 | 3529 | 19420 |
| factorization time | 3.18 | 9.5 | 30 |

Tolerance \( 10^{-3} \)

| 3-set (average) iterations | 21.80 | 17 | 15.95 |
| 3-set (average) solve time | 0.75 | 1.38 | 9.15 |
| time-split. iterations | 250 | 241 | 389 |
| time-split. solve time (single thread) | 1880 | 10023 | 215888 |
| time-split. solve time (N threads*) | 188 | 334.1 | 3598 |

Tolerance \( 10^{-3} \)

| 3-set (average) iterations | 13.14 | 13.27 | 12.32 |
| 3-set (average) solve time | 0.49 | 1.18 | 7.34 |
| time-split. iterations | 156 | 128 | 224 |
| time-split. solve time (single thread) | 780 | 4304 | 99525 |
| time-split. solve time (N threads*) | 78 | 143.47 | 1659 |

* estimated parallel computation times
The dual updates are
\[ \tilde{w}_t^{k+1} = \tilde{w}_t^k - G_0 \tilde{x}_t^{k+1} + z_t^{k+1}, \quad t = 1, \ldots, T. \]
(27)
\[ \tilde{v}_t^{k+1} = \tilde{v}_t^k - G_1 \tilde{x}_t^{k+1} + z_t^{k+1}, \quad t = 1, \ldots, T. \]
(28)
Substituting (27) and (28) into (26) results in \( \tilde{w}_t^{k+1} + \tilde{v}_t^{k+1} = 0 \), for all \( t \). Hence \( z_t^{k+1} \) can be simplified to
\[ z_t^{k+1} = \frac{G_0 \tilde{x}_t^{k+1} + G_1 \tilde{x}_t^{k+1}}{2}, \quad t = 1, \ldots, N. \]
This result shows that \( z_t \) is the variable that enforces consensus upon \( x_t^{(i)} \) and \( x_t^{(j-1)} \), the value of which is the average of the two.

**Derivation of the three-set splitting algorithm updates**

The augmented Lagrangian for (28) can be written as
\[ L_\alpha(x, z, y, \bar{z}, \bar{y}, v) := \frac{1}{2} x^T M x + q^T x + r + v^T (\tilde{x} - y) \]
\[ - \rho \sum_{i} \sum_{k} z_i \| x_i - y_i \|^2 \]
\[ - \rho \sum_{k} \| \tilde{z}^k - \tilde{y}^k \|^2. \]
As before, the dual variable \( v \) is associated with the equality constraints. We define the indicator function of the nonnegative orthant for a variable \( x \in \mathbb{R} \) as
\[ I_{x \geq 0} = x \begin{cases} 1 & x \geq 0 \\ \infty & \text{otherwise} \end{cases} \]
The function applies componentwise for vector variables.

The augmented Lagrangian is a smooth, quadratic function with respect to the variables \( x^I, x^H, x^ III \), hence the updates can be derived from taking the gradient equal to zero. This yields the solutions (16), (17), and (18).

Grouping the terms that involve \( z \) from (29) we obtain
\[ z_t^{k+1} = \arg\min_z \left( \sum_{i} \frac{\rho}{2} \| z_i - (x_i^{k+1} + z_i^{k}) \|^2 \right) \]
\[ = \frac{1}{3} \sum_{i} (x_i^{k+1} + z_i^{k}). \]
(30)
Maximizing (29) over \( \bar{z}_i \), \( i = I, II, III \) results in
\[ (\bar{z}_i^{k+1}) = \arg\max_{\bar{z}} \left( \frac{\rho}{2} \| \bar{z}_i \|^2 + \frac{\rho}{2} \| \bar{z} - (x_i^{k+1} + z_i^{k+1}) \|^2 \right). \]
Hence, the update for \( \bar{z}_i \), \( i = I, II, III \) is
\[ (\bar{z}_i^{k+1}) = (\bar{z}_i)^k - (x_i^{k+1} + z_i^{k+1}), \quad i = I, II, III. \]
(31)
Similarly, the update for \( \bar{y} \) is
\[ y^{k+1} = y^{k} + y^{k+1} - h + H(\tilde{x}^{(III)}^{k+1}). \]
Taking the sum of the dual variables \( \bar{z}_i \), \( i = I, II, III \) from (31) and substituting into (30) simplifies the \( z \)-update and we obtain
\[ z_t^{k+1} = \frac{1}{3} \sum_{i} (x_i^{k+1}). \]
Hence, the global variable \( z \) is the average of the three estimates of \( x \).
[22] P. Amestoy, T. Davis, and I. Duff, “Algorithm 837: AMD, an approximate minimum degree ordering algorithm,” ACM Transactions on Mathematical Software, vol. 30, pp. 381–388, Sept. 2004.

[23] T. Davis, Direct Methods for Sparse Linear Systems. SIAM Fundamentals of Algorithms, Society for Industrial and Applied Mathematics, 2006.