2008

Multiplexed Predictive Control of a Large Commercial Turbofan Engine

Hanz Richter
Cleveland State University, h.richter@csuohio.edu

Anil V. Singaraju
Cleveland State University

Jonathan S. Litt
U.S. Army Research Laboratory

Original Citation

Richter, H., Singaraju, A. V., and Litt, J. S., 2008, "Multiplexed Predictive Control of a Large Commercial Turbofan Engine," Journal of Guidance, Control, and Dynamics, 31(2) pp. 273-281.

Multiplexed Predictive Control of a Large Commercial Turbofan Engine

Hanz Richter* and Anil Singaraju†
Cleveland State University, Cleveland, Ohio 44115
and
Jonathan S. Litt‡
U.S. Army Research Laboratory, Cleveland, Ohio 44135

Model predictive control is a strategy well-suited to handle the highly complex, nonlinear, uncertain, and constrained dynamics involved in aircraft engine control problems. However, it has thus far been infeasible to implement model predictive control in engine control applications, because of the combination of model complexity and the time allotted for the control update calculation. In this paper, a multiplexed implementation is proposed that dramatically reduces the computational burden of the quadratic programming optimization that must be solved online as part of the model-predictive-control algorithm. Actuator updates are calculated sequentially and cyclically in a multiplexed implementation, as opposed to the simultaneous optimization taking place in conventional model predictive control. Theoretical aspects are discussed based on a nominal model, and actual computational savings are demonstrated using a realistic commercial engine model.

I. Introduction

AUTONOMOUS propulsion control of aircraft engines is being pursued based on the promise of higher fuel efficiency, extended component life, better transient response, and better robustness to engine-to-engine variations and engine aging [1]. Relieving the overall workload of the pilot has been another driver for pursuing this technology. Model predictive control (MPC) is a technique that has reached maturity in proven applications as well as in theoretical foundations. MPC has been demonstrated to have the potential to achieve the preceding goals in the context of aircraft propulsion.

In this paper, we present a technique for reducing the complexity of MPC laws, so that real-time implementation is feasible with limited computational resources. Focus is placed on the application of MPC to aircraft engines, in which practical implementation of the predictive laws is problematic, due to limited onboard processing power and memory. The multiplexed MPC (MMPC) implementation proposed here greatly reduces the time required for computing the control law, with small performance degradation. The article is organized as follows: Section II describes the mathematical model for the engine, along with an overview of MPC development, applications, computational burden, and a brief description of alternate measures to reduce it. Application of MPC to aircraft engine control problems is also described. In Sec. III, the concept of multiplexing is introduced, along with theoretical results for nominal linear plants, including the use of an observer. Section IV describes the application of the multiplexed scheme to the actual engine model and presents simulation results demonstrating computational savings with little performance deterioration. Finally, Sec. V offers conclusions.

II. Engine Modeling and Predictive Control

A. Mathematical Model

Aircraft engine dynamic models are characterized as being highly complex and nonlinear. As customarily done in model-based control design, a relatively simple model must be used in control law derivations, reserving high-fidelity, high-complexity models for the validation stage. A design model is used in this paper that correctly maps the influence of actuators upon outputs and also provides a large number of other variables of interest.

The design model is still nonlinear and complex to a high degree. The state equations are not available in closed form, because algebraic loops requiring iterative solutions are present. Further, the algebraic constraints also involve lookup tables. These characteristics imply that the state derivatives may only be evaluated numerically. The dynamic model of the turbofan engine considered here can be written as

$$\dot{x} = f(x, z, u, p) \tag{1}$$

$$0 = g(x, z, u, p) \tag{2}$$

$$y(x) = h(x, u, z, p) \tag{3}$$

where x is the state vector, u is the control vector, z are the variables participating in the algebraic loops, and p is a vector of uncertain and/or time-varying parameters related to engine health and aging. The output vector y contains a number of variables of interest, among them the thrust developed by the engine and the turbine inlet temperature, which will be controlled. The control vector reflects the four available actuators: namely, the fuel flow rate, the variable bleed-valve opening, the variable stator-vane opening, and the turbine blade clearance. In this paper, only the first three are used for feedback control.

The model is essentially a differential-algebraic system, a type that has been extensively studied [2,3]. However, it is the numerical nature of $f(.)$, $g(.)$, and $h(.)$, together with the dependence on the uncertain parameters p, that makes the controls problem very difficult to solve using many of the standard analytical tools. Basic properties such as controllability, observability, and open-loop stability of the preceding model are not possible to establish in a rigorous fashion by conventional methods.

Aside from the difficulties imposed by the structure of the model, dimensionality also poses a major problem. There are more than 90 states and more than 20 algebraic variables, together with four available actuators. Merely simulating the model in an open-loop fashion requires extensive software and specialized routines. As explained in [4], the open-loop functions $h(.)$ and $g(.)$ are linearized to provide a Newton-based scheme to iteratively solve for the algebraic variables to a specified level of accuracy at each time step. Specifically, if z_n and y_n are available from the previous step, together with current values of p, u, and x, the next value of z can be approximated as

$$z_{n+1} = L^{-1}(y_{n+1} - y_n) + z_n$$

where $L = \partial g/\partial z$, evaluated at z_n and the current values of u, x, and p. Ideally, iteration for z should be continued until a very small value is obtained in the evaluation of $g(.)$, and L should be a function of u, x, and p. In practice, it is found that a fixed number of iterations and a constant value for L are sufficient to provide a good estimate of z. Once z has been estimated, it is used in the main numerical integration routine for x.

B. Overview of Model-Predictive Control

MPC represents a paradigm shift with respect to more traditional approaches. Well-known control laws such as proportional–integral, state feedback, and transfer function compensation are based on observations of past and current errors. That is, a nonzero error must have been measured for a control action to occur. MPC, in contrast, uses a mathematical model of the process to predict the errors that would be incurred if certain control inputs were to be applied. It then selects the control inputs that minimize the predicted errors. This strategy is performed continually, as new information is made available to the controller. Thus, MPC offers the possibility of reformulating the way it computes the next control move, according to the newest information about the plant and its external influences (reference commands and disturbance). Figures 1 and 2 illustrate the usual MPC scheme.

At time instant k, an internal plant model and a disturbance model, together with measurements of current plant outputs and commanded references, are used to obtain a function $\hat{y}$, which gives predicted values of the output over a time range N_p as a function of future control inputs over another time interval N_u. Usually, $N_u \leq N_p$ and the control values are assumed constant beyond time $N_u + k$. The function $\hat{y}$ is used in the minimization of a measure of the deviations of predicted outputs from the reference inputs r over a time interval $[k, k + N_p]$. The measure to be minimized is usually a weighted quadratic sum of output deviations

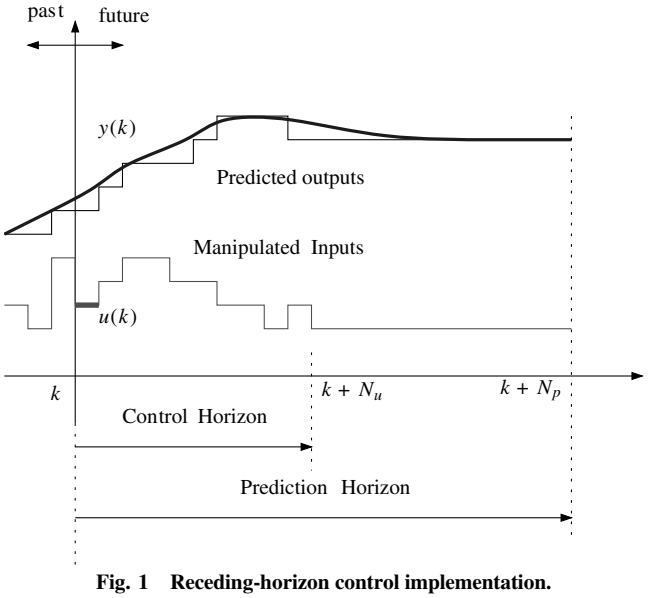

Fig. 1 Receding-horizon control implementation.

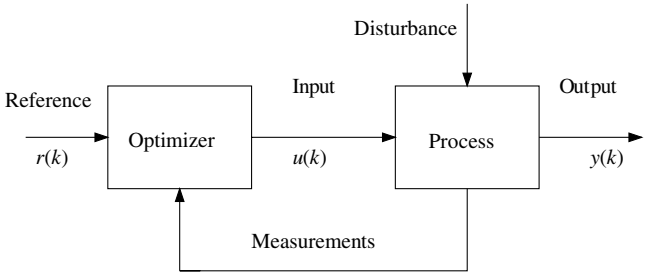

Fig. 2 General MPC structure.

and control penalties, although other measures are possible and were considered. The optimization problem yields a sequence of N_u control values achieving minimal deviations, usually under control and state constraints. Only the first control move in the sequence is made effective by the actuators during sampling interval k. The entire process is repeated at every sample instant, resulting in an enhanced ability to handle constraints, reject disturbances, and follow the reference trajectory. Irrespective of the exact form of the objective function being optimized at each sampling instant and the nature or existence of constraints, this implementation is universally known as *receding-horizon control*. It is evident that the complexity of the computations for a given form of the objective function depends on the order of the model, the length of the prediction and control horizons, the number of inputs, the number of outputs, and the number and nature of the constraints. In this paper, we address computational complexity by reducing the number of inputs that are to be determined at each time step by the online optimization process by introducing a multiplexed implementation.

MPC has come to designate a variety of related control techniques that have in common the online optimization process, the use of a plant model, and the receding-horizon implementation. Receding-horizon ideas can be traced back to the 1960s, but active interest in the field started in the 1980s with the introduction of dynamic matrix control [5] and generalized predictive control [6–8]. Predictive control strategies were extensively and successfully tested in real-world process control applications, sparking even greater interest from researchers. Despite the growing popularity of the technique, solid theoretical analyses and stability proofs appeared only in the 90s, with the seminal works of Mayne et al. [9–11], Bemporad and Morari [12], Rawlings and Muske [13], and Bitmead and Gevers [14], among others. Theoretical works also started to appear that were applicable to nonlinear systems. Until the 90s, the applicability of MPC was confined to "slow" systems such as oil refining and other chemical processes, due to the large amount of time required to perform the optimization calculations. Currently, due to rapid progress in developing fast and cheap computing power, MPC is being considered as a serious candidate for faster processes such as mechanical systems and aircraft engine controls. Theoretical MPC research in the last five years seems directed to complexity reduction by piecewise implementation [15], multiplexed schemes [16–18], and to hybrid approaches that can optimize over logical or integer variables. For a survey containing industrial applications, see [19]. For a historical survey of theoretical developments and stability proofs, see [9]. It is worth mentioning that MPC has matured to the point of commercialization by companies such as Honeywell and ABB.

C. MPC for Aircraft Propulsion

The application of MPC to aircraft engines has become a realistic option with the availability of faster onboard processing [20,21]. MPC represents a marked departure from conventional turbofan engine control schemes, which typically regulate fan speed or engine pressure ratio, because those are the measured variables that are most closely related to thrust. MPC is inherently multivariable, taking advantage of actuators that are not used for feedback purposes in conventional schemes, but are simply scheduled. MPC automatically handles disturbances as well as input and state constraints. For a

survey of modern control technology as it applies to engine control and health monitoring, see [22]. The MPC scheme addressed a variety of tasks better handled by model-based computation than by pilots: notably, the compensation for engine deterioration over time and the generation of thrust trajectories that reduce fuel consumption. Automatic fault detection and accommodation are features being integrated in the MPC formulation via recursive parameter estimation. Recently, MPC has been applied to active turbine blade clearance control [21].

In this section, the model and the current MPC strategy are described. Although nonlinear versions of MPC exist that guarantee stability and can work with constraints [10,11], they are not applicable to our engine control problem without a number of simplifications and assumptions. The approach taken is reasonable in view of problem constraints. The state equation (1) is linearized every sampling instant, using the estimated values of z from the Newton approach. This operation generates a linear continuous-time model of the form

$$\dot{x} = \hat{A}x + \hat{B}_u u + \hat{B}_p p \tag{4}$$

$$y = \hat{C}x + \hat{D}_u u + \hat{D}_p p \tag{5}$$

Note that the model is really time-varying, because the values of the system matrices are updated at the beginning of each sampling instant. It is also important to note that analytical gradients for f and h are not available, thus requiring a numerical perturbation approach at every sampling instant to obtain the linearized system matrices. Because the health parameters p are slowly varying, they can be considered to be a bias term for the purposes of MPC computation. Next, the preceding model is discretized using a zero-order-hold in the inputs, leading to a discrete-time version of Eqs. (4) and (5). The predictions for MPC are incorporated through the discretized model matrices. The MPC optimization problem has the form

$$\min_{u_k, u_{k+1}, \ldots u_{k+N_u}} J = \sum_{i=1}^{N_p} (r_{k+i} - \hat{y}_{k+i})^T Q (r_{k+i} - \hat{y}_{k+i}) + \sum_{i=0}^{N_u} \Delta u_{k+i}^T R \Delta u_{k+i} \tag{6}$$

subject to state and control constraints (rate and magnitude).

The degrees of freedom in the optimization are the control increments $\Delta u_{k+i} = u_{k+i} - u_{k+i-1}$ for $i = 1, 2, \ldots, N_u$. The reference inputs are r_k, assumed to be known, and $\hat{y}(k)$ is a model-dependent prediction vector. For details regarding predictors, refer to [23,24]. Matrices Q and R are positive definite and symmetric, as customarily used in control optimization. If the state vector is available for measurement and if the constraints are absent, the preceding optimization problem has a closed-form solution, depending on system matrices. The resulting control law has the form of a state feedback, in which the gain can be readily computed from the current measurements. When constraints are present, more general and powerful quadratic programming (QP) algorithms are needed. The state vector and, moreover, the parameter vector p are unknown. For this reason, an extended Kalman filter was tuned to provide state and parameter estimates to the MPC optimization, in a fashion analogous to observer-based control design. The Kalman gain and the covariance matrix are not constant, but rather updated at every sampling instant. Simulation results show that little performance is lost when the MPC is implemented based on state estimates [4].

D. Computational Burden of MPC

The domain of applicability of MPC has reached beyond slow-processing control-oriented problems. The availability of fast and cheap real-time processing has motivated work on MPC applied to aircraft engines [20,25]. Onboard processors used in aircraft propulsion control are still not powerful enough to accommodate the execution of the complex algorithm in real time. In the following sections, a brief overview is offered on MPC variants specifically designed to reduce the computational cost.

1. Explicit MPC Implementations

In an explicit MPC implementation, the solution of the optimization problem inherent to predictive control is performed offline. The solution consists in the computation of regions of the state space and corresponding control gains to be used in the actual real-time implementation. An explicit MPC implementation is essentially a multidimensional lookup table for the control gain. This approach requires that the plant model and objective function be reducible to a multiparametric programming problem. That is, the optimization problem should have the initial state appearing as a parameter in a linear fashion. The paper by Bemporad et al. [26] develops a performance criterion based on the sum of either the ∞-norm or 1-norm of the input command and the deviation of the state from its desired value. By taking the ∞-norm over space and the 1-norm over time, it is possible to reduce the problem to a multiparametric case. The solution offered by using the mixed $1/\infty$-norm is attractive for relatively small linear MPC problems. This method saves computing time by offering precomputed solutions at each time step. In the case of large plants, using this approach would impose high demands in memory because of the high number of state variables. In addition, the explicit implementation is not well-suited to plants exhibiting high variability, and the possibility of reducing the problem to multiparametric programming is difficult to establish.

2. 1-Norm Criterion with End Condition

Dynamic matrix control (DMC) involves a system description in terms of step-response parameters. This description is used in a receding-horizon optimization to generate a sequence of controls that minimizes the deviations between the actual and desired step responses. DMC is traditionally used in conjunction with a quadratic objective function (QDMC). As seen in [28], an objective function based on the 1-norm can be used along with an end condition. This approach is computationally less intensive than QDMC. By introducing an end condition in the performance index, it is possible to obtain a stable and high-performance control system even when using input/output constraints and short prediction horizons. The control law is calculated by solving an online linear program, which is less complex than a quadratic program.

3. Min–Max Approaches

A cost function is introduced in [29] that allows the formulation of a robustly stable MPC problem solvable by a linear program. Using a $1/\infty$-norm performance index for this cost allows precomputation of the solution, so that the linear program does not have to be solved online. The objective of predictive control is to compute the future control sequence $u(k), u(k+1), \ldots, u(k+N_u)$ in such a way that the optimal j step-ahead prediction $y(k+j|k)$ is driven close to $r(k+j)$ for the prediction horizon.

The preceding three approaches focus on changing the objective function so that the optimization problem becomes linear. In some cases, this allows part or all of the optimization process to be moved offline. Our approach differs in that it uses the familiar quadratic optimization problem, which is well-known and for which many efficient numerical recipes are available. Computational savings arise from the reduced dimensionality of the online optimization. In fact, the time required to solve a QP problem grows with the cube of the number of inputs. We show that our multiplexed approach can be implemented without significant performance deterioration.

III. Overview of Multiplexed Control

A *multiplexed control implementation* denotes an arrangement in which a group of actuators is updated sequentially and cyclically, as opposed to simultaneously. In this technique, only a group of actuators are updated every sampling instant, keeping all other actuators held at their previous values. The group of actuators being

updated is given by a predetermined schedule. A multiplexing schedule specifies the instants at which each actuator is commanded to the value dictated by the control law. This technique can be implemented in software, hardware, or in a combination of both, and has applications in various fields of engineering.

The exact physical residence of the multiplexer depends on the purpose for its implementation. In electronics, the multiplexer is a hardware element inserted between the group of signals and the hardware resources to be shared. In controls, one may use a multiplexer for the same reasons as in electronics [for instance, to save on the number of signal-conditioning channels and data converters (hardware multiplexing)] or the multiplexer may be implemented in software, becoming an integral part of the control law (software multiplexing). Multiplexed control has received recent attention in the context of MPC [16–18], typically restricting the number of actuators being updated at every sampling instant to one. In the following sections, we use a nominal linear model to discuss the theoretical implications of introducing multiplexing in conjunction with conventional linear quadratic control.

A. Some Theoretical Results for Nominal Linear Plants

Consider a discrete-time, linear, time-invariant plant in state-space form:

$$x(k+1) = Ax(k) + Bu(k) \tag{7}$$

where A and B have dimensions n by n and n by m, respectively; $x(k) \in \mathbb{R}^n$; and $u(k) \in \mathbb{R}^m$. To introduce integral action in the control law and facilitate modeling of the multiplexed implementation, the control input $u(k)$ is assumed to have the form

$$u(k) = u(k-1) + \Delta u(k) \tag{8}$$

where $\Delta u(k)$ is the control increment to be optimized by the MPC law. The state vector is augmented with the m components of $u(k-1)$, resulting in the description

$$\begin{bmatrix} x(k+1) \\ u(k) \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & I_m \end{bmatrix} \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} + \begin{bmatrix} B \\ I_m \end{bmatrix} \Delta u(k) \tag{9}$$

Denoting the extended state vector as $\tilde{x}(k) = [x(k)^T | u(k-1)^T]^T$ and the new system matrices as A_g and B_g, the augmented system becomes

$$\tilde{x}(k+1) = A_g \tilde{x}(k) + B_g \Delta u(k) \tag{10}$$

Assume that q groups of r control inputs each are to be simultaneously updated, where, for simplicity, $q = m/r \in \mathbb{Z}^+$ and $r < m \leq n$. Without loss of generality, assume that the q update groups are contiguous in Δu and that the update sequence coincides with the order in which the groups are stacked in Δu. A simple input transformation can produce the assumed arrangement starting from any other configuration satisfying $m = qr$. Define the *p-dimensional selector matrix at time k* as the q-periodic matrix:

$$E_p(k) = \left[{}_p e^T_{r(k \bmod q)+1} \middle| {}_p e^T_{r(k \bmod q)+2} \middle| \cdots \middle| {}_p e^T_{r(k \bmod q)+r} \right]$$

with ${}_p e_j$ being the jth canonical basis row vector of $\mathbb{R}^p$. The multiplexed plant can now be described as

$$\tilde{x}(k+1) = A_g \tilde{x}(k) + b_g(k) w(k) \tag{11}$$

where

$$b_g(k) = B_g E_m(k)$$

Clearly, $b_g(k)$ is periodic with period q, and $w(k)$ is an r vector reparameterizing the control inputs according to

$$w(k) = E_m^T(k) \Delta u(k) \tag{12}$$

Note that the selector matrices satisfy the property $E_m^T(k) E_m(k) = I_r$, where I_r is the r by r identity matrix. Also, the product $E_m(k) E_m^T(k)$ gives an m by m block-diagonal matrix, where the blocks are I_r and an $(m-r)$ by $(m-r)$ zero matrix. Therefore, Eq. (13) may be written as

$$\Delta u(k) = E_m(k) w(k) \tag{13}$$

The preceding correspondence provides a mechanism to model the assignment of zero values to selected components of Δu outside their scheduled update instants.

B. Assumptions and Basic Definitions

Several basic definitions and assumptions must be introduced before the control derivations. Applying multiplexing to a plant introduces periodicity to the closed-loop system. Multiplexing is equivalent to periodically setting all columns of B_g to zero, except for a block of columns corresponding to the actuators being updated at a particular instant. Therefore, the plant has a periodically varying input distribution matrix. Control design must take this into account, because the time-invariant eigenvalue condition for stability does not apply. The *monodromy matrix* of $A(k)$ at time j is defined as

$$\Phi_A(T+j, j) \triangleq A(T+j-1)A(T+j-2), \ldots, A(j)$$

that is, the product of all instances of $A(k)$ over one period. The eigenvalues of Φ_A are known as *characteristic multipliers*. These eigenvalues are independent of j. A periodic system $[A(k), B(k)]$ is said to be *stabilizable* if a T-periodic feedback matrix $F(k)$ can be found such that $A(k) - B(k)F(k)$ is asymptotically stable. The periodic system $[A(k), B(k)]$ is *asymptotically stable* if and only if all characteristic multipliers lie in the open disc $\{z \in \mathbb{C}: |z| < 1\}$. The *periodic Kalman decomposition* with periodic components is given as

$$\begin{bmatrix} z_{11}(k+1) \\ z_{12}(k+1) \end{bmatrix} = \begin{bmatrix} A_{uc}(k) & 0 \\ A_{21}(k) & A_c(k) \end{bmatrix} \begin{bmatrix} z_{11}(k) \\ z_{12}(k) \end{bmatrix} + \begin{bmatrix} 0 \\ B_c(k) \end{bmatrix} \Delta u(k) \tag{14}$$

A periodic system $[A(k), B(k)]$ is stabilizable if the matrix associated with the uncontrollable subspace in a periodic Kalman decomposition is asymptotically stable. For instance, as discussed in [27], every periodic pair $[A(k), B(k)]$ admits a canonical decomposition analogous to the usual Kalman controllability form, but with periodic block components. Because the situation will arise in which a pair $[A, B(k)]$ is made up of a constant A and a q-periodic $B(k)$, we interpret the constant matrix as periodic with period q, with the peculiarity $A(k) = A$ for all k. Thus, the period is not defined here as the smallest T for which $A(k+T) = A(k)$ for all k, but rather a predetermined value of T for which the equality holds. Such arrangement extends the definition of and conditions for stabilizability for such constant-periodic pairs.

Assumption 1: In connection with system (7), define the periodic matrix $b(k) \triangleq BE_m(k)$.

1) The pair $[A, b(k)]$ is stabilizable.

2) Rank $[b(k)] = r$ for all k.

Note that the constant pairs $[A, b(k_1)]$ for some fixed k_1 need not be stabilizable; that is, a constant state-feedback gain might not exist that asymptotically stabilizes the system when restricted to a single update group. The stabilizability assumed for $[A, b(k)]$, however, guarantees that a periodic state-feedback gain exists that stabilizes the reduced system (and the original system, by virtue of the proposed technique), as will be shown in the following sections. An immediate consequence of Assumption 1 is that the pair $[A_g, b_g(k)]$ is stabilizable and rank$[b_g(k)] = r$ for all k.

In fact, augmentation with input integrators introduces m unity eigenvalues in A_g, which are associated with the equation

$$u(k) = u(k-1) + E_m(k) w(k)$$

The portion of the augmented state vector corresponding to $u(k-1)$ is decoupled from $x(k)$ and fully controllable from $w(k)$, as a simple rank calculation shows. Therefore, the newly introduced eigenvalues

belong to the controllable subspace of $[A_g, b_g(k)]$, showing stabilizability. It is straightforward to show that $\text{rank}[b_g(k)] = r$ for all k under Assumption 1.

C. Infinite-Horizon Quadratic Regulator

An infinite-horizon quadratic regulator for the multiplexed plant of Eq. (11) is obtained from the minimization of

$$J = \sum_{k=0}^{\infty} \tilde{x}^T(k) Q \tilde{x}(k) + w^T(k) R w(k)$$

The solution is well known [30] to be a periodic state feedback of the form

$$w(k) = -F(k)\tilde{x}(k) \tag{15}$$

Computation of $F(k)$ is done by solving the system of discrete-time periodic Riccati equations:

$$X_j = Q + A_g^T(j) X_{j+1} A_g(j) - A_g^T(j) X_{j+1} b_g(j) \Big[R + b_g^T(j) X_{j+1} b_g(j) \Big]^{-1} b_g^T(j) X_{j+1} A_g(j)$$

for $j = 0, 1, 2, \ldots, (q-1)$. Under the stabilizability assumption, a unique, symmetric, and positive-semidefinite periodic sequence X_j can be found. The desired periodic feedback gain is then computed from

$$F_j = -\Big[R + b_g^T(j) X_{j+1} b_g(j) \Big]^{-1} b_g^T(j) X_{j+1} A_g(j)$$

Note that the q equations cannot be solved separately and that the solution method can be described as numerically intensive. Several methods exist to solve the preceding system of Riccati equations [31]. In one of them, an initial symmetric, positive-semidefinite, and stabilizing X_0 is calculated from a forward-time discrete periodic Lyapunov equation (FTDPLE). In turn, several methods are available for solving the FTDPLE. One of them reduces to the solution of a standard discrete Lyapunov equation. Once X_0 is available, a Newton step having the form of a reverse-time discrete periodic Lyapunov equation (RTDPLE) is repeated until convergence of X_j to a periodic sequence. More recent, computationally efficient, methods involving the periodic Schur decomposition are also described in [31,32]. The reader is referred to these works and references therein for a detailed exposition of numerical methods. The actual control input $u(k)$ is found through Eqs. (8) and (13). Multiplexing can be applied to systems that are not stabilizable from the update group of actuators if taken individually. That is, systems that are not stabilizable when some actuators are permanently deactivated, but that are stabilizable in the periodic sense, can be multiplexed.

D. Multiplexed Control with Observer

The multiplexed plant of Eq. (11) has an augmented state vector in which the last m components are the previous value of the control input: that is, $u(k-1)$. A full-order observer designed on the basis of this plant would produce an estimate of u, introducing redundancy. Using a reduced-order observer can remove this redundancy. The reduced-order observer presented here estimates plant states only, and the augmented state vector is formed by stacking these estimates and the actual values of $u(k-1)$, which are assumed to be stored in the control algorithm. As shown in Sec. III, introduction of multiplexing to a plant results in a periodic system. Even though the system being observed is periodic in nature, the observer designed in this case is a linear time-invariant system. This system is stabilized independently using a feedback gain that drives the error between state and estimate to zero. We now offer a new result that can be understood as a version of the well-known *separation principle* for observer-based multiplexed control.

The plant to be estimated is the same as Eq. (7), with the addition of an output vector of dimension p:

$$x(k+1) = Ax(k) + Bu(k) \tag{16}$$

$$y(k) = Cx(k) \tag{17}$$

The estimator update equation is given as

$$\hat{x}(k+1) = (A - HC)\hat{x}(k) + Bu(k) + Hy(k) \tag{18}$$

As before, the control input $u(k)$ is assumed to have the following form:

$$u(k) = u(k-1) + \Delta u(k) \qquad \Delta u(k) = E_m(k) w(k)$$
$$w(k) = -F(k)\tilde{x}(k)$$

where

$$\tilde{x}(k) = \begin{bmatrix} \hat{x}(k) \\ u(k-1) \end{bmatrix}$$

The equation for the error dynamics of the system is obtained by taking the difference between the estimated state $\hat{x}(k)$ and the actual state $x(k)$. As customary, the error dynamics are independent of the control law being used. Define

$$e(k) = \hat{x}(k) - x(k) \tag{19}$$

From Eqs. (16) and (18), we get

$$e(k+1) = (A - HC)e(k) \tag{20}$$

The closed-loop dynamics of the multiplexed plant under linear state feedback and using the estimated values of the state from the observer can be derived in a straightforward manner. From Eq. (8), we have

$$\Delta u(k) = -E_m(k) F(k) \begin{bmatrix} \hat{x}(k) \\ u(k-1) \end{bmatrix}$$

Substituting the preceding control law into the plant equation and using Eq. (19), we obtain

$$x(k+1) = Ax(k) + Bu(k-1) - BE_m(k)F(k) \begin{bmatrix} x(k) + e(k) \\ u(k-1) \end{bmatrix} \tag{21}$$

After some matrix algebra [33], it is possible to reduce the closed-loop dynamics to

$$\begin{bmatrix} \bar{x}(k+1) \\ e(k+1) \end{bmatrix} = \begin{bmatrix} A - b_g(k)F(k) & b_g(k)F(k) \\ 0 & A - HC \end{bmatrix} \begin{bmatrix} \bar{x}(k) \\ e(k) \end{bmatrix} \tag{22}$$

where

$$\begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} \equiv \bar{x}(k)$$

The block-triangular structure of the preceding system matrix implies that plant and estimator may be stabilized separately. Note that the periodic stabilizability of (A, b_g) was established earlier.

E. Multiplexing in Receding-Horizon Control

Application of multiplexing to receding-horizon strategies is straightforward. At each time step, a constrained MPC problem of reduced dimension is solved, corresponding to a subset of all available actuators. Only the first value of the calculated control sequence is applied, whereas the remaining actuators are kept at their previous values. The problem is solved again at subsequent time steps, but using the appropriate subset of actuators. In [17], two related algorithms are presented, assuming full state information. In one of the schemes, all actuators are optimized at once, but under the constraint that their increment will be nonzero only every m time steps, where m is the number of actuators. This scheme does not

reduce the dimensionality of the constrained optimization routine, and so it is of little advantage in terms of computational efficiency. In the second scheme, only one actuator at a time is optimized. Information about the previously predicted values of the other actuators is included in the optimization. The approach amounts to treating the nonoptimized actuators as known disturbances. The predicted values of the actuators will never be realized, however, due to the receding nature of the control algorithm. The simulations presented in this paper do not use information about the values of the nonoptimized actuators, but this information can certainly be incorporated in a straightforward fashion.

The main concern in a multiplexed MPC implementation is stability. The closely related periodic estimation problem has been analyzed by DeNicolao [34]. The concept is to seek conditions under which a periodic extension of the first few values of the optimal gain sequence can be stabilizing when used as periodic feedback. These conditions are obtained by considering the cyclomonotonicity properties of the discrete periodic Riccati equation (DPRE). In [35], a similar problem is considered and sufficient conditions for stability are derived. These works can be used as a basis to derive theoretical stability conditions for the nominal multiplexed plant of Eq. (11) under receding-horizon control. These derivations are works in progress.

IV. Application of Multiplexed MPC to the Turbofan Engine Model

The motivation for including a multiplexer is to reduce the complexity (i.e., dimensionality) of MPC calculations. At sampling instant k, linearization and discretization are performed to derive a single-input linear model, in which the input is selected from among the set of all actuators according to a predefined cyclic schedule. All other inputs are assumed constant and equal to the values they had at sampling instant $k-1$. The single-input linear model is used in the MPC optimization of Eq. (6), resulting in an optimal move sequence for the selected input. The first element of the sequence is applied to the corresponding plant actuator, whereas the other actuator commands are held at their previous values. The operation is repeated at the following sampling instants for the remaining inputs one by one and according to the schedule, completing what will be termed an *update cycle*. Update cycles are repeated indefinitely, until the control system is stopped. The update cycle is illustrated in Fig. 3. Note that the effective sample rate was reduced to the original one divided by the number of actuators. However, the rate at which *some* actuator is being updated is the same as the original.

The computational advantage of the multiplexed implementation lies in the fact that all QP routines are now performed over just one

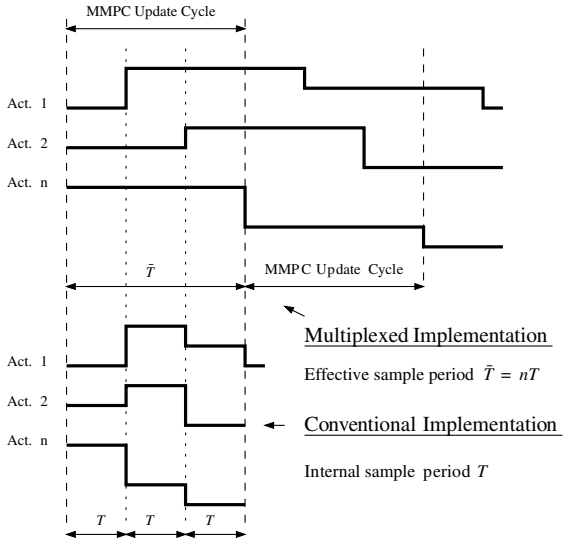

Fig. 3 Multiplexed control updates.

degree of freedom. It is a well-established fact that the time required to solve a QP problem grows with the cube of the number of inputs [36], whereas the sample rate reduction is only linear in the number of inputs. Therefore, the time savings earned by MMPC may even allow an increase of the original sample rate to help recover any lost performance due to slower sampling. This is especially true for the disturbance rejection properties (a faster rate helps reduce the effects of disturbance in the intersample). This possibility is heavily dependent on the problem at hand, because the other computational costs need to be taken into account.

A. Implementation of the MMPC Control Law

Multiplexing was implemented on three of the four available actuators: namely, fuel flow rate, variable stator-vane opening (VSV), and variable bleed-valve opening (VBV). Fuel is optimized and updated at "modulo 3" sampling instants (0, 3, 6, etc.). Actuator VBV is optimized and updated at "modulo 3 plus 1" sampling instants (1, 4, 7, etc.) and actuator VSV at "modulo 3 plus 2" sampling instants (2, 5, 8, etc.). Two engine outputs provided by Eq. (3) were controlled: thrust and turbine inlet temperature. Traditionally, thrust is controlled indirectly using the fan speed or engine pressure ratio, but the onboard model provides an estimate of the actual thrust for control. Turbine inlet temperature is controlled because it has a large impact on component life, and so it is important that it is maintained within an acceptable range. By adjusting the actuators using MPC, the engine can be controlled such that the optimal thrust response is obtained while minimizing specific fuel consumption and preserving part life.

The linear model of Eqs. (4) and (5) is obtained by linearization with respect to these inputs and outputs. A number of constraints on the states and the inputs are present in the optimization. With this multiplexing approach on the engine model, we demonstrate substantial computational savings while achieving almost the same performance for the thrust as obtained with MPC. As will be seen next, the savings increase dramatically for large prediction horizons, starting with 27% corresponding to control and prediction horizons of 2. The computation times reported here were obtained by timing single quadratic optimization function calls and averaging the results during the course of a simulation. Therefore, these times are independent of the way code was written and provide meaningful comparison with the conventional MPC.

B. Simulation Results

The following simulations illustrate the performance of MMPC in comparison with conventional MPC. Simulations shown here correspond to multiplexing of all three actuators controlling the two outputs after suitably retuning the system for the required performance in thrust. In fact, significant performance loss was observed when implementing the multiplexed approach with the original Q and R weights used in the original MPC implementation. Retuning to recover the performance in thrust was done manually, by trial-and-error simulations. Simulations correspond to a takeoff thrust profile, which is used as a benchmark for control system performance.

As can be seen in Fig. 4, the thrust response is similar to that obtained from MPC, but Fig. 5 shows some jitter in the turbine inlet temperature response. However, further tuning may be able to remove this effect. Exact-temperature tracking is difficult to accomplish with the three selected actuators. Instead of attempting to track a temperature profile, leaving the temperature as a constraint might prove more useful in future studies. In addition, the use of nonupdated actuator values as known biases in the online optimization process might help improve the response. Figures 6–8 show the control input update for fuel, VBV, and VSV resulting from optimization performed for each actuator in a multiplexed fashion. Note that the control trajectories for VSV and VBV are significantly different from those of the original MPC. These simulations are based on the fixed value of control and prediction horizons (both equal to 20). Figure 9 shows the thrust response of MMPC system for various values of control and prediction horizons. In this figure, ch

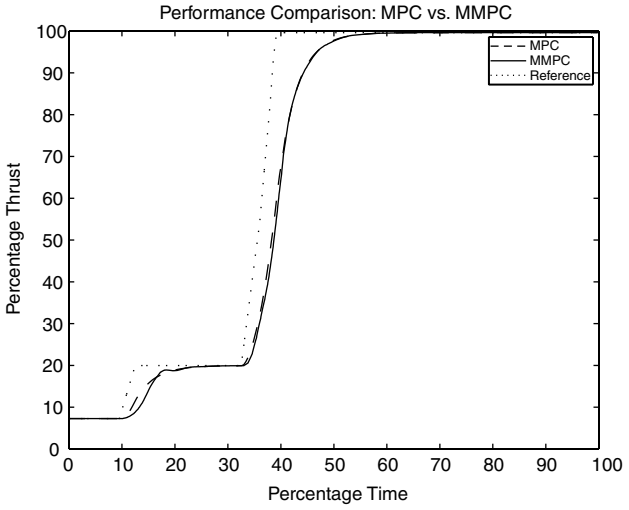

Fig. 4 Thrust comparison: MPC vs MMPC.

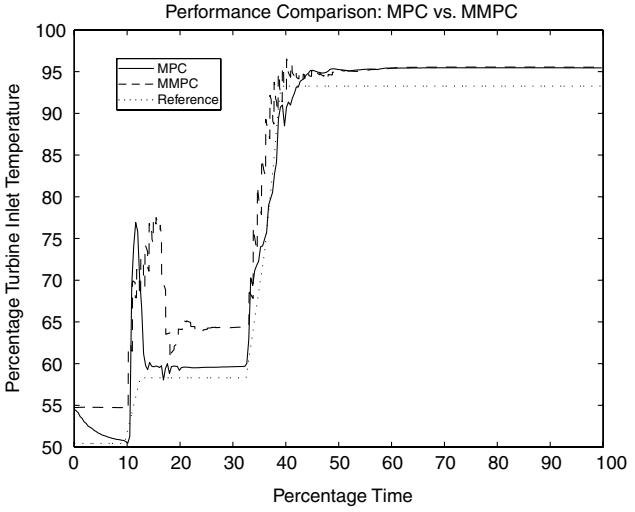

Fig. 5 Temperature comparison: MPC vs MMPC.

represents the value of the control horizon and ph represents the prediction horizon. During the up-transient, all horizon combinations result in approximately the same performance, except for the case ch = ph = 2, which gives a slightly longer rise time. Finally, Fig. 10 shows the comparison of MPC and MMPC in terms of the

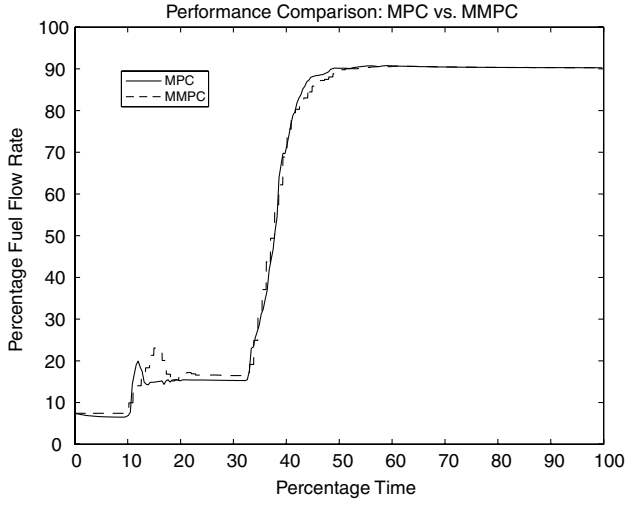

Fig. 6 Fuel-consumption comparison: MPC vs MMPC.

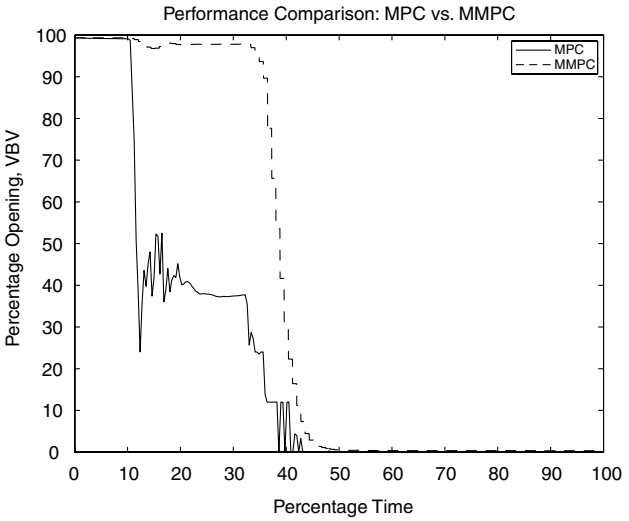

Fig. 7 Variable bleed-valve opening comparison: MPC vs MMPC.

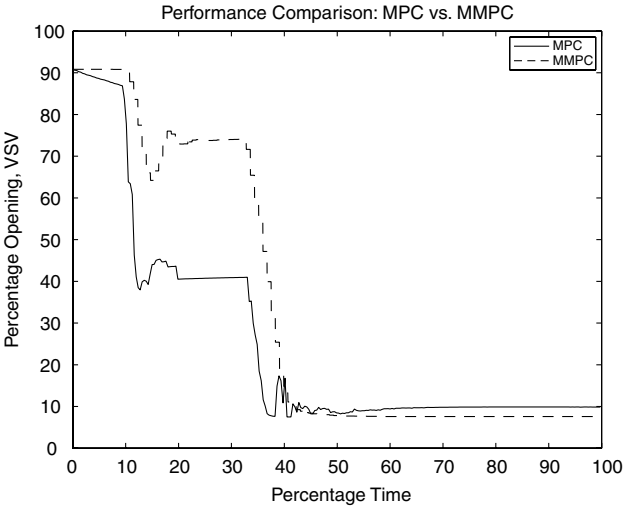

Fig. 8 Variable stator-valve opening comparison: MPC vs MMPC.

computational time taken by each of them in calculating the optimal control law with variation in control and prediction horizon.

Simulations show that the thrust response of the engine using MMPC is similar to that obtained with MPC after some tuning, even

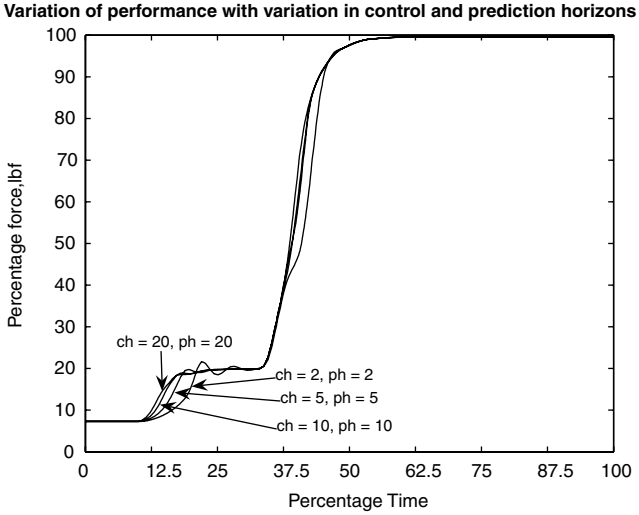

Fig. 9 Comparison of MMPC with variation in prediction horizon (ph) and control horizon (ch).

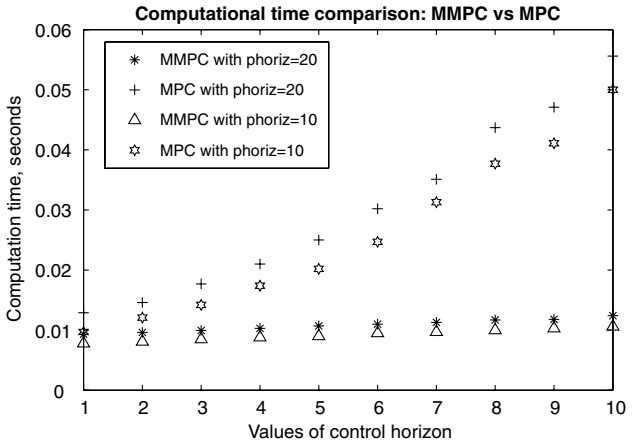

Fig. 10 Comparison for computation time: MMPC vs MPC.

though the turbine temperature response shows some jitter. This provides reassurance that the system will not be destabilized and that it is likely to meet other performance requirements for the engine by systematic tuning of the systems. It is important to note that the ultimate purpose is real-time implementation. The bottleneck lies in the computation of the actuator updates. If the time taken for this computation plus overhead calculations is larger than the sampling period permitted by the processor, real-time implementation is not feasible. MMPC updates only one actuator at a time, greatly reducing the computation time and enabling the use of slower processing. For horizons of 8 and larger, MMPC is faster by a factor of 3 or more.

V. Conclusions

The MPC technique represents a significant improvement over traditional engine control approaches. Transient performance optimization, disturbance rejection, and constraint handling are combined in one systematic approach to control design. High-level objectives such as pilot workload reduction, fuel economy improvement, and component life extension can all be incorporated as part of a control objective to be solved in the framework of MPC.

However, MPC is characterized by a large computational cost, which precludes real-time implementation. The use of multiplexing in MPC laws reduces the computational time required for calculating the optimal control law by reducing the dimensionality of the quadratic program. This can be done without adversely affecting performance of the closed-loop system if retuning is performed. The work presented here could be the basis for conducting test-stand trials with actual engines and processors. The time savings earned by MMPC is large enough to allow an increase in the original sample rate. This may be used to help recover any lost performance due to the slower effective rate introduced by multiplexing, in particular, in terms of disturbance rejection. The use of an observer with the multiplexed approach does not present difficulties. We have also shown that the nominal multiplexed plant can be estimated by a conventional linear time-invariant observer and that the usual separation principle holds.

Acknowledgment

This work was supported by the NASA John H. Glenn Research Center at Lewis Field in Cleveland, Ohio, through grant NNC05GA73G.

References

[1] Garg, S., "Controls and Health Management Technologies for Intelligent Aerospace Propulsion Systems," NASA John H. Glenn Research Center at Lewis Field TM-2004-212915, Cleveland, OH, 2004.

[2] Kumar, A., and Daoutidis, P., "Feedback Control of Non-Linear Differential Algebraic Equations," *AIChE Journal*, Vol. 41, No. 3, 1995, pp. 619–636. doi:10.1002/aic.690410319

[3] Rhem, A., and Allgöwer, F., "General Quadratic Performance Analysis and Synthesis of Differential-Algebraic (DAE) Systems," *Journal of Process Control*, Vol. 12, No. 4, 2002, pp. 467–474. doi:10.1016/S0959-1524(01)00013-0

[4] Kumar, A., and Viassolo, D., "Autonomous Propulsion System Technology," GE Global Research, Rept. NAS 3-01135, Schenectady, NY, 2004.

[5] Cutler, C. R., and Ramaker, B. L., "Dynamic Matrix Control—A Computer Control Algorithm," Joint Automatic Control Conference, American Automatic Control Council, Dayton, OH, Paper WP5-B, 1980.

[6] Clarke, D. W., Mohtadi, C., and Tuffs, P. C., "Generalized Predictive Control, Part 1: The Basis Algorithm," *Automatica*, Vol. 23, No. 2, 1987, pp. 137–148. doi:10.1016/0005-1098(87)90087-2

[7] Clarke, D., Mohtadi, C., and Tuffs, P. C., "Generalized Predictive Control, Part 2: Extensions and Interpretations," *Automatica*, Vol. 23, No. 2, 1987, pp. 149–160. doi:10.1016/0005-1098(87)90088-4

[8] Bordons, C., and Camacho, E., "A Generalized Predictive Controller for a Wide Class of Industrial Processes," *IEEE Transactions on Control Systems Technology*, Vol. 6, No. 3, 1998, pp. 372–387. doi:10.1109/87.668038

[9] Mayne, D. Q., Rawlings, J. B., Rao, C. V., and Scokaert, P. O. M., "Constrained Model Predictive Control: Stability and Optimality," *Automatica*, Vol. 36, No. 6, 2000, pp. 789–814. doi:10.1016/S0005-1098(99)00214-9

[10] Mayne, D., and Michalska, H., "Receding Horizon Control of Nonlinear Systems," *IEEE Transactions on Automatic Control*, Vol. 35, No. 7, 1990, pp. 814–824. doi:10.1109/9.57020

[11] Michalska, H., and Mayne, D., "Robust Receding Horizon of Constrained Linear Systems," *IEEE Transactions on Automatic Control*, Vol. 38, No. 11, 1993, pp. 1623–1633. doi:10.1109/9.262032

[12] Bemporad, A., and Morari, M., *Robust Model Predictive Control: A Survey*, edited by A. Garulli, A. Tesi and A. Vicino, Vol. 245, Lecture Notes in Information and Control Sciences, Springer, New York, 1999, pp. 207–226.

[13] Rawlings, J., and Muske, K., "The Stability of Constrained Receding Horizon Control," *IEEE Transactions on Automatic Control*, Vol. 38, No. 10, 1993, pp. 1512–1516. doi:10.1109/9.241565

[14] Bitmead, R., and Gevers, M., *Adaptive Optimal Control*, Prentice–Hall, Upper Saddle River, NJ, 1990.

[15] Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N., "The Explicit Quadratic Regulator for Constrained Systems," *Automatica*, Vol. 38, No. 1, 2002, pp. 3–20. doi:10.1016/S0005-1098(01)00174-1

[16] Cagienard, R., Grieder, P., Kerrigan, E. C., and Morari, M., "Move Blocking Strategies in Receding Horizon Control," *Proceedings of the 43rd IEEE Conference on Decision and Control*, Vol. 2, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 2004, pp. 2023–2028.

[17] Ling, K. V., Maciejowski, J. M., and Wu, B.-F., "Multiplexed Model Predictive Control," 16th IFAC World Congress, International Federation of Automatic Control Paper We-M09-TO/6, 2005.

[18] Sun, J., Chen, S., and Kolmanovsky, I., "A Stable Block Model Predictive Control with Variable Implementation Horizon," *Proceedings of the 2005 American Control Conference*, Vol. 2, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 2005, pp. 834–839. doi:10.1109/ACC.2005.1470063

[19] Qin, S. J., and Badgwell, T. A., "An Overview of Industrial Model Predictive Control Technology," *Chemical Process Control-V*, Computer Aids for Chemical Engineering Education (CACHE), Austin, TX, 1997, pp. 232–256.

[20] Brunell, B., Bitmead, R., and Connolloy, A., "Nonlinear Model Predictive Control of an Aircraft Gas Turbine Engine," *Proceedings of the 41st IEEE Conference on Decision and Control*, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 2002, pp. 4649–4651.

[21] DeCastro, J., "Rate-Based Model Predictive Control of Turbofan Engine Clearance," 42nd AIAA/ASME/ASME/ASEE Joint Propulsion Conference and Exhibit, AIAA Paper 2006-5108, 2006.

[22] Litt, J., Simon, D. L., Garg, S., Guo, T.-H., Mercer, C., Millar, R., Behbahani, A., Bajwa, A., and Jensen, D. T., "A Survey of Intelligent Control and Health Management Technologies for Aircraft Propulsion

Systems," *Journal of Aerospace Computing, Information, and Communication*, Vol. 1, No. 12, 2004, pp. 543–563.

[23] Camacho, E., and Bordons, C., *Model Predictive Control*, Springer, New York, 2003, pp. 13–18.

[24] Rossiter, J., *Model-Based Predictive Control*, CRC Press, Boca Raton, FL, 2003, pp. 31–52.

[25] Fuller, J., Kumar, A., and Millar, R., "Adaptive Model-Based Control of Aircraft Propulsion Systems: Status and Outlook for Naval Aviation Applications," American Society of Mechanical Engineers, Paper GT2006-90241, 2006.

[26] Bemporad, A., Borrelli, F., and Morari, M., "Model Predictive Control Based on Linear Programming—The Explicit Solution," *IEEE Transactions on Automatic Control*, Vol. 47, No. 12, 2002, pp. 1974–1985.
doi:10.1109/TAC.2002.805688

[27] Bittanti, S., and Bolzern, P., "Stabilizability and Detectability of Linear Periodic Systems," *Systems and Control Letters*, Vol. 6, No. 2, 1985, pp. 141–145.
doi:10.1016/0167-6911(85)90083-0

[28] Genceli, H., and Nikolau, M., "Robust Stability Analysis of Constrained l_1 Norm Model Predictive Control," *AIChE Journal*, Vol. 39, No. 12, 1993, pp. 1954–1965.
doi:10.1002/aic.690391206

[29] Kerrigan, E. C., and Maciejowski, J. M., "Robustly Stable Feedback Min-Max Model Predictive Control," *Proceedings of the 2003 American Control Conference (ACC 2003)*, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 2003, pp. 3490–3495.

[30] Bittanti, S., Colaneri, P., and De Nicolao, G., *The Periodic Riccati Equation*, edited by S. Bittanti, J. A. Laub and J. C. Willems, Springer, New York, 1991, pp. 127–162.

[31] Varga, A., "Periodic Lyapunov Equations: Some Applications and New Algorithms," *International Journal of Control*, Vol. 67, No. 1, 1997, pp. 69–87.
doi:10.1080/002071797224360

[32] Sreedhar, J., and Van Dooren, P., "On Finding Stabilizing State Feedback Gains for a Discrete-Time Periodic System," *Proceedings of the 1994 American Control Conference*, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, 1994, pp. 1167–1168.

[33] Singaraju, A., "Multiplexed Control of Hardware and Computationally Constrained Systems," M.S. Thesis, Cleveland State Univ., Cleveland, OH, 2006.

[34] De Nicolao, G., "Cyclomonotonicity and Stabilizability Properties of Solutions of the Difference Periodic Riccati Equation," *IEEE Transactions on Automatic Control*, Vol. 37, No. 9, 1992, pp. 1405–1410.
doi:10.1109/9.159582

[35] Yan, W., and Bitmead, R. R., "Periodic Receding Horizon LQ Regulators for Discrete-Time Systems," *Proceedings of the 30th IEEE Conference on Decision and Control*, Inst. of Electrical and Electronics Engineers, Piscataway, NJ, Dec. 1991, pp. 2301–2306.
doi:10.1109/CDC.1991.261573

[36] Rao, C., Wright, S. J., and Rawlings, J. B., "Application of Interior-Point Methods to Model Predictive Control," *Journal of Optimization Theory and Applications*, Vol. 99, No. 3, 1998, pp. 723–757.
doi:10.1023/A:1021711402723