Nonlinear model predictive missile control with a stabilising terminal constraint

V. Bachtiar * T. Mühlpfordt ** W. H. Moase *
T. Faulwasser *** R. Findeisen ** C. Manzie *

* Department of Mechanical Engineering
The University of Melbourne, Australia
{bachtiarv,moasew,manziec}@unimelb.edu.au
** Otto-von-Guericke-Universität Magdeburg, Germany
tillmann.muehlpfordt@st.ovgu.de, rolf.findeisen@ovgu.de
*** École Polytechnique Fédérale de Lausanne, Switzerland
timm.faulwasser@epfl.ch

Abstract: In this paper, an MPC scheme for a missile pitch axis autopilot is proposed. The scheme uses a nonlinear prediction model to give it an ability to push the controlled missile very close to its operating limits, and is stabilised through the use of an ellipsoidal terminal constraint. Tracking performance and computational load of the scheme are compared to that with a linear prediction model and other types of terminal constraint. Specifically, the choice of ellipsoidal, polytopic or no terminal constraint is discussed. The terminally constrained nonlinear MPC scheme achieves comparable solution times to that with a linear prediction model, whilst being more aggressive to give a superior tracking performance.

Keywords: Predictive control, nonlinear control, missiles, stability

1. INTRODUCTION

Model Predictive Control (MPC) has a number of attractive properties for the control of a guided airframe. It explicitly handles constraints and has the ability to directly take into account plant nonlinearities. These are crucial as MPC would be able to push a missile to operate near its physical limits, at high angles of attack where aerodynamics are highly nonlinear (Gros et al., 2012). The other components of MPC such as objective cost and prediction horizon can be formulated to achieve the most desirable control behaviour. This makes MPC suited to address current challenges in missile autopilot design, which often revolve around the inability to account for nonlinearities, changes in missile behaviour during flight, and different missile configurations (Jackson, 2010).

Despite the aforementioned advantages, applications of MPC for guiding missiles are rare. This is primarily due to the high computational demand of MPC which can be problematic for applications in areas where plants posses fast, nonlinear dynamics, such as in the case of a missile (Hu and Chen, 2007). This makes the trade-off between theoretical advantages and implementability of MPC for missile control an important discussion.

To address the issue regarding computational burden, typical approaches include simplification of the problem formulation through model linearisation, relaxation of constraints, or bounding by uncertain linear models. Although reducing computational cost, these approximations can potentially have detrimental effects on the closed loop performance of the controller, such as a steady-state error due to model-plant mismatch or even instability.

The prediction model dictates the complexity of the pertaining optimisation problem in MPC. A nonlinear prediction model is associated with a higher computational load than a linear model due to the nonconvex nature of the resulting optimisation problem for most real-world applications. Early methods in solving MPC with a nonlinear prediction model utilised the direct multiple-shooting (Bock and Plitt, 1984) or collocation (Biegler, 1984). To achieve computational times needed in missile applications, a fast algorithm involving a sequential convex programming (Tran-Dinh and Diehl, 2010) to solve the nonlinear MPC scheme is followed in this paper. For benchmarking, the prediction model is simplified by a typical linearisation.

In this paper, a terminal constraint is used to guarantee stability of the MPC. In the early derivations of MPC stability, the terminal constraint was used to impose the terminal state to coincide at an invariant point, the origin (e.g. Mayne and Michalska, 1990). This is rather restrictive and in subsequent developments the notion of an invariant region (rather than an invariant point) was introduced (Michalska and Mayne, 1993). Chen et al. (1998; 2001) extended the theory by approximating the invariant region as an ellipsoid. An alternative approach in approximating such a region is to use a polytope (Cannon et al., 2003), which could better approximate the invariant region. These approaches derive closed-loop MPC stability with a linear differential inclusion (LDI) approximation of the nonlinear plant model, which simplifies calculation of the terminal region at the expense of being conservative. This paper compares the ellipsoid and polytopic terminal...
region for missile autopilot. Further, these two approaches are compared to a computationally simpler approach with a relaxed constraint, i.e. the use of no terminal region.

The comparisons of the prediction models and terminal constraints are on the basis of how quickly the controller can drive the missile to track a given acceleration command and its computational load. This is to establish an understanding of the quality of nonlinear MPC with an ellipsoidal terminal constraint in terms of tracking performance and implementability of achieved solution times.

2. PLANT MODEL

2.1 Missile pitch-axis dynamics

The missile autopilot control in the pitch-axis as depicted in Fig. 1 is governed by the equations
\[ \dot{q} = \frac{L(\alpha, \delta)}{I_y} \]
where the aerodynamic lift force \( F_L \) and pitching moment \( L \) are modelled by (Nichols et al., 1993)
\[ F_L = 0.7M^2P_0S(C_{Z\alpha,1}(2 - M/3)\alpha + C_{Z\alpha,2}\alpha \alpha + C_{Z\delta}\delta) \]
\[ L = 0.7M^2P_0Sd[C_{Lo,1}(8M/3 - 7)\alpha + C_{Lo,2}\alpha \alpha + C_{Lo,3}\delta + C_{Lo}\delta] \]
\[ \alpha \] is the angle of attack and \( q = \dot{\varphi} \) is the pitch rate of the missile. The output of the system \( \alpha \) is the normal acceleration of the missile in multiples of gravitational acceleration \( g \). Missile speed \( V = MV_s \), where \( V_s \) is the speed of sound, is treated as constant at Mach number \( M = 2.5 \). The actuation of the fin deflection \( \delta \) is modelled as a second order system:
\[ \ddot{\delta} = -\omega_\delta^2\delta - 2\omega_\delta^2\dot{\delta} + \omega_\delta^2\delta \cdot (3) \]
Aerodynamic coefficients \( C_{Z\alpha,1}, C_{Z\alpha,2}, \) etc., other flight condition and missile frame parameters, along with constants related to (3) used are the same as given in Nichols et al. (1993). Note that the symbols \( C_{Z\alpha,1}, C_{Z\alpha,2} \) etc. follow the standard nomenclature, and are different to that used by Nichols et al. (1993).

The dynamics of the missile can be put in a concise form:
\[ \begin{align*}
\dot{x} &= f(x, u) \\
y &= h(x)
\end{align*} \quad (4a)\]
with state variables \( x = [\alpha, q, \dot{\alpha}, \dot{\delta}, \delta]^T \in \mathbb{R}^{n_x} \), control variables \( u = \dot{\delta} \in \mathbb{R}^{n_u} \), and output \( y = \alpha \).

The missile operation is subject to a number of constraints. A constraint on \( \alpha \) is associated with the fact that aerodynamic coefficients used are only valid for a range of \( \alpha \). Fin deflection \( \delta \) and its rate \( \dot{\delta} \) are subject to mechanical limits of the actuator. Although there is no physical restriction that limits \( q, \dot{\alpha} \), a constraint is imposed (made large for it to never be active) to make compact constraint polytopes
\[ X = \{ x : -\bar{x} \leq \dot{x} \leq \bar{x} \} \cup U = \{ u : -\bar{u} \leq u \leq \bar{u} \} \]
(5) where \( \leq \) and \( \geq \) denote element-wise inequalities. This describes a region within missile physical limitations and accuracy of aerodynamic coefficients. The states and input limits are \( \bar{x} = [\bar{\alpha}, \bar{q}, \bar{\dot{\alpha}}, \bar{\delta}, \bar{\delta}]^T \), and \( \bar{u} = \bar{\delta} \) respectively. Due to these constraints, the missile is only capable of tracking a maximum acceleration of \( a_{max} \).

2.2 Tracking problem formulation

The autopilot control receives a commanded normal acceleration \( a_o \) to track from the missile guidance law. Desired steady-state values of state variables \( x_o \) which achieve \( a_o \) are obtained from the solution of \( 0 = f(x_o, u_o) \) and \( a_o = h(x_o) \). The system is injective therefore a commanded normal acceleration \( a_o \) is associated with one unique steady state \( x_o = [\alpha_o, q_o, \delta_o, 0, \delta_o]^T \), \( u_o = 0 \).

If \( a_o \neq 0 \) then \( x_o \neq 0 \). The system can be formulated as an error system with an equilibrium at the origin \( x = 0 \):
\[ \begin{align*}
\dot{x} &= \tilde{f}(\tilde{x}, u) = f(x, u) - f(x_o, u_o) = \tilde{f}(x, u) \\
X &= \{ x : -\bar{x} \leq x < x_o \leq \bar{x} \}, U = \{ u : -\bar{u} \leq u \leq \bar{u} \} \quad (6b)
\end{align*} \]

3. MODEL PREDICTIVE CONTROL

The missile in continuous time \( t \) is controlled at each sampling time \( t_k \), for \( i = 1, 2 \ldots \) separated by a sampling period \( T_s \), i.e. \( t_{i+1} = t_i + T_s \). At each sampling instant, with current state \( x(t_i) \), the MPC scheme considered in this paper is to solve the optimisation problem
\[ \min_{x_k, u_k = 1, \ldots , N} \sum_{k=1}^{N} \ell(x_k, u_k) + e(x_{N+1}) \quad (7a) \]
s.t. \[ \begin{align*}
x_k &= x(t_i) \\
x_{k+1} &= \Phi(x_k, u_k) \\
x_k &\in X, u_k \in U \\
x_{N+1} &\in X_f
\end{align*} \quad (7b) \]
\[ \Phi(x_k, u_k) = x_k + \int_{t_k}^{t_{k+1}} f(x(\tau), u_k) \ d\tau \] \quad (8)

The subscript \( k \) is used to discretise the continuous variables \( x \) and \( u \) into \( N \) discrete prediction variables to be solved by computational means. Here, \( N \) characterises the prediction horizon of the MPC scheme. The prediction model \( \Phi(x_k, u_k) \) takes an initial state \( x_k \) and integrates the tracking error model (6b) over one sampling period with zero-order hold input \( u_k \) to obtain a predicted state \( x_{k+1} \). The solution of (7a–e) are the optimal state \( x_k \) and control sequence \( u_k^* \), \( \forall k = 1 \ldots N \), the first of which, \( u_1^* \), is applied as feedback control to the plant.

3.1 Cost function

The cost function (7a) (with stage cost \( \ell(\cdot) \) and terminal cost \( e(\cdot) \)) is a performance measure of the missile within the prediction horizon, indicating how far the states are to the desired values. A typical quadratic cost function:
\[ \ell(x, u) = \| x \|^2_Q + \| u \|^2_R \quad , \ e(x) = \| x \|^2_P \] \quad (9)
is used. \( Q \) and \( R \) are positive definite stage cost weighting matrices for the state and control variables respectively, and \( P \) is the terminal cost weighting matrix. Minimising (9) drives the error \( x \) to zero and the system states \( \hat{x} \) to the steady state \( x_s \) corresponding to the command \( a_s \).

To only penalise errors in states relevant to the acceleration, the following formulation is used for \( Q \) and \( R \):

\[
Q = C^T C \quad \text{where} \quad C = \frac{\partial h}{\partial x} \quad \text{and} \quad R = \epsilon R \tag{10}
\]

noting that \( \|x\|^2_{Q,x_s} \approx (a - a_s)^2 \). This eliminates the need for tuning of \( Q \) and \( R \), and simplifies the design process of the controller. Because the input \( u \) is constrained, there is no need to penalise its deviations from zero. \( R \) is set to \( \epsilon R \) where \( 0 < \epsilon R < 1 \) to ensure that (7a–e) is locally convex with respect to \( u \), and avoid a chattering control behaviour. The value of \( P \) is chosen to ensure stability and will be found using different approaches in Section 3.3.

3.2 Prediction model

To solve (7a–e) with the nonlinear prediction model, equations (7b–c) can be formulated as a dynamic system representing the difference between the states \( \hat{x} \) and an estimate of their values \( \hat{x}_s \), \( \forall k = 1 \ldots N \), represented by a first-order Taylor expansion along the estimated trajectories:

\[
x_{k+1} = \Phi(\hat{x}_k, \hat{u}_k) + \frac{\partial \Phi}{\partial x} \bigg|_{\hat{x}_k, \hat{u}_k} (x_k - \hat{x}_k) + \frac{\partial \Phi}{\partial u} \bigg|_{\hat{x}_k, \hat{u}_k} (u_k - \hat{u}_k).
\]

(11)

Defining the Jacobians and a new vector \( F_k \) at each time step in the horizon as

\[
A_k = \frac{\partial \Phi}{\partial x} \bigg|_{\hat{x}_k, \hat{u}_k}, \quad B_k = \frac{\partial \Phi}{\partial u} \bigg|_{\hat{x}_k, \hat{u}_k}, \quad (12a)
\]

and

\[
F_k = -\Phi(\hat{x}_k, \hat{u}_k) + A_k \hat{x}_k + B_k \hat{u}_k \quad (12b)
\]

the differences between the true and estimated states and controls can now be captured in the following inequality constraint, replacing (7b–c) in the optimisation problem:

\[
\begin{bmatrix}
B_1 - I \\
A_2 B_2 - I \\
\cdots \\
A_N B_N - I \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\vdots \\
u_N \\
\end{bmatrix}
= 
\begin{bmatrix}
F_1 - A_1 x_1 \\
F_2 \\
\vdots \\
F_N \\
\end{bmatrix}.
\]

(13)

Following Tran-Dinh and Diehl (2010), the overall nonlinear MPC (NMPC) algorithm to solve (7a–e) is:

**Algorithm 1.** [NMPC] \((k = 1 \ldots N + 1 \text{ for } x \text{ and } k = 1 \ldots N \text{ for } u) \)

0 Obtain the current state \( x(t_i) \). Initialise the estimate: \( \hat{x}_k = x(t_i) \), \( \hat{u}_k = 0 \). Set \( j = 0 \).

1 Get an optimal \((x_k, u_k)\) by solving (7a) s.t. (7b,7d–e,11). Denote this optimal solution \((\hat{x}_k, \hat{u}_k)\), \( j = j + 1 \).

2 Compute normalised mean squared errors: \( \Delta x = \frac{1}{N+1} \sum_{k=1}^{N+1} ((\hat{x}_k - x_k) / \hat{x}_k)^2 \), \( \Delta u = \frac{1}{N} \sum_{k=1}^{N} ((\hat{u}_k - u_k) / \hat{u}_k)^2 \).

3 If \( \Delta x < \epsilon x \) and \( \Delta u < \epsilon u \), or \( j > j_{\text{max}} \) for some thresholds \( \epsilon x \ll 1, \epsilon u \ll 1 \), and \( j_{\text{max}} \), then go to 4. Otherwise re-initialise: \( \hat{x}_k = \hat{x}_k, \hat{u}_k = \hat{u}_k \) and go to 1.

4 Apply control \( \hat{u}_k \). Move forward in time: \( i = i + 1 \). Note that at this step, \( \hat{u}_i \approx u_i \) (as per Remark 1 below).

5 Obtain \( x(t_i) \). To hot-start Steps 1–3, re-initialise the estimate: \( \hat{x}_k = \hat{x}_k, \hat{u}_k = \hat{u}_k \). Then shift: \( \hat{x}_{k+1} = \hat{x}_k, \hat{u}_{k+1} = \hat{u}_k \). Set \( j = 0 \). Go to 1.

In Step 1, (7a) s.t. (7b,7d–e,11) is solved as a quadratic program (QP). Then Steps 2–3 effectively impose the condition \( x_{k+1} = \Phi(x_k, u_k) \) for (11), ultimately satisfying (7c). In Step 3, the condition \( j < j_{\text{max}} \) can be used to ensure that the optimisation is solved under a certain time.

**Remark 1.** [Convergence of Algorithm 1 (Theorem 1 in Tran-Dinh and Diehl (2010))] \((\hat{x}_k, \hat{u}_k)\) in Steps 1–3 of Algorithm 1 converges to \((x_k^*, u_k^*)\) if \((\hat{x}_k, \hat{u}_k)\) is initialised sufficiently close to \((x_k^*, u_k^*)\).

**Linear prediction model**

Algorithm 1 yields an exact solution of (7a–e). As a benchmark, a computationally simpler alternative which yields an approximate solution of (7a–e) can be achieved by a time-invariant linearisation of (7c) about the reference \((\hat{x} = x_0)\), i.e. origin of the error system \((x = 0)\), yielding a linear MPC scheme (LMPC) with

\[
x_{k+1} = \frac{\partial f}{\partial x} \bigg|_0 x_k + \frac{\partial f}{\partial u} \bigg|_0 u_k = Ax_k + Bu_k \tag{14}
\]

In place of (11). Now in (13), \( A_k = A, B_k = B, \) and \( F_k = 0, \forall k = 1 \ldots N \) in place of (12a–b). Consequently, (7a–e) can be solved as a QP in one step, simplifying the computation of \( u^* \) at each sampling instant, although introducing a greater model error in approximating (7c).

3.3 Terminal constraint

The terminal constraint \( x_{N+1} \in X_f \), along with the stage cost \( f(\cdot) \), terminal cost \( e(\cdot) \), and prediction horizon \( N \) are responsible for the stability of the MPC scheme. Specifically, to follow Mayne and Rawlings (2009) closely:

**Theorem 1.** [MPC Stability (Mayne and Rawlings, 2009)]

For continuous \( f(\cdot), \ell(\cdot), e(\cdot) \), and compact \( X \) and \( U \), the closed-loop error system with (7a–e) is asymptotically stable about the origin if \( X_f \) is a subset of \( X \) and \( X \), where \( X \) is defined as a control invariant region where once the system state \( x \) enters, there is a terminal controller \( x = \kappa(x) \in U \) such that \( \Phi(x, \kappa(x)) \) stays within \( X_a \).

The methods of computing \( X_f \) followed in this paper are derived on a linear differential inclusion (LDI) approximation of the nonlinear missile model, which is more conservative and easier to analyse. The terminal controller considered is a linear controller \( \kappa(x) = Kx \) for some gain matrix \( K \). An associated control Lyapunov function (CLF) with \( \kappa(x) \) can be determined once \( x \) enters \( X_s \).

**No terminal constraint**

The conditions in Theorem 1 are sufficient but not necessary for closed-loop stability. Guarantee of stability without a terminal constraint has been shown by approximating a stable combination of \( P \) and \( N \) (Parisini and Zoppoli, 1995) for example. Grüne and Pannek (2011) use controllability assumptions which are in general very challenging to verify for a given nonlinear system. Thus, for a given \( N \), trial-and-error is employed for \( P \) when no terminal constraint is used.

**Ellipsoidal terminal constraint**

In this paper, a terminal constraint (7e) is used to guarantee stability through The-
Theorem 1. To keep (7e) simple for computational purposes, \( X_f \) will be restricted to either a polytope or ellipsoid. With a \( C_1 \) model
Comparing NMPC and LMPC, both schemes achieve zero steady-state error and no overshoot for a step change

\[ X_f = \mathcal{E} = \{ x : x^T W x \leq \gamma \} \]

(15)

for some \( W \) and \( \gamma \) is a common choice, with a volume of \( \text{vol}(\mathcal{E}) = 4/3\pi \text{det}(W/\gamma) \). This paper follows Chen et al.’s (2001) proposed stable NMPC with \( X_f = \mathcal{E} \), with a terminal cost weighting \( P = W \). The associated CLF is

\[ \| \Phi(x, Kx) \|_P^2 - \| x \|_P^2 \leq -l(x, Kx) \].

(16)

For volume maximisation of \( \mathcal{E} \) under state and input constraints, and such that \( X_f \subseteq \mathcal{X}_s \), Chen et al.’s approach through linear matrix inequalities is used in this paper.

**Polytopic terminal constraint** As an alternative to an ellipsoid, a polytope

\[ X_f = \Pi = \{ x : \| V x \|_\infty \leq \lambda \} \]

(17)

for some \( V \) and \( \lambda \) can be chosen as the terminal region, with a volume of \( \text{vol}(\Pi) = (2\lambda)^n/\text{det}(V) \). Cannon et al.’s (2003) work for stability under such a terminal region and its volume optimisation is followed in this paper, with \( P = Q \) and an associated CLF of

\[ \| V \Phi(x, Kx) \|_\infty^2 - \| x \|_\infty^2 \leq 0 \].

(18)

Cannon et al.’s first method (Theorem 3 and Remark 5) is used for the volume optimisation of \( \Pi \) under state and input constraints, and \( X_f \subseteq \mathcal{X}_s \). The motivation for \( X_f = \Pi \) is that it may lead to a volumetrically larger \( X_f \) than with \( X_f = \mathcal{E} \) depending on the exact shape of \( \mathcal{X}_s \). This comes at a cost; the two methods for volume maximisation of \( \Pi \) by Cannon et al. are nonconvex. They are associated with multiple local minima, unlike the convex, globally-optimum method for \( \mathcal{E} \).

**Tracking of non-constant references** As the reference \( a_0 \) changes, the constraint polytope \( X \) in (6c) changes, and so does the region \( \mathcal{X}_s \). As a result the terminal constraint must be recalculated. Recalculation of the terminal region \( X_f \) is computationally expensive and is not feasible to do within each sampling period. A simple approach is to precompute a set of \( X_f \) for a finite number of acceleration commands \( a_L = \{ a_1 \ldots a_{n_L} \} : \]

\[ \mathcal{E}_l = \{ x : x^T W_l x \leq \gamma \} \quad l = 1 \ldots n_l \].

(19)

In this subsection, \( \mathcal{E} \) is interchangeable with \( \Pi \). \( n_l \) is the number of precomputed terminal regions. \( l = 1, 2 \ldots n_l \) corresponds to \( a_0 = a_1, a_2 \ldots a_{n_l} \), in increasing order from \( a_0 = 0 \) to \( a_{n_l} = a_{\text{max}} \). Note that \( X_f \) for \( a_0 = a_{\text{max}} \) contains only the origin. The terminal region can be recalculated with (the system is symmetric about the origin thus an extension for a negative acceleration command is trivial):

**Assumption 1.** [Strictly shrinking \( X_f \)] \( \mathcal{E}_{l+1} \subseteq \mathcal{E}_l \), \( \forall l_2 > l_1 \).

**Proposition 1.** [Stability for set-point changes] Consider the following method for determining the terminal constraint for a given \( a_0 \notin a_{L} \); Find \( a_0 \) and \( a_l \), the smaller and larger immediate neighbours of \( a_0 \) in \( a_{L} \) respectively, with the associated terminal regions \( \mathcal{E}_l \) and \( \mathcal{E}_l' \). Set the terminal constraint with \( X_f = \mathcal{E}_l' \). For continuous \( f(\cdot), l(\cdot) \), \( e(\cdot) \), compact \( X \) and \( U \), and satisfaction of Assumption 1, the method (extended accordingly for \( a_0 < 0 \)) leads to asymptotic stability of the closed-loop error system in tracking set-point reference changes to \( a_0 \notin a_L \) for \( -a_{\text{max}} \leq a_0 \leq a_{\text{max}} \).

**Remark 2.** Assumption 1 is made on the basis that as \( a_0 \) increases, the constraint polytope \( X \) shifts away from the origin. Thus, since \( \mathcal{E} \) is symmetric about the origin, a smaller terminal region is required to keep \( \mathcal{E}_l \subseteq X \). If Assumption 1 is valid, \( \mathcal{E}_l \subseteq \mathcal{E}_l' \) in Proposition 1. The method given in the proposition is then conservative in guaranteeing stability with a more restrictive constraint.

**Remark 3.** [Stability for time-varying references] Theorem 1 and Proposition 1 are valid for set-point references. For time-varying references i.e. trajectory tracking, the error system has been shown to converge to a bounded region under certain assumptions (Falugi and Mayne, 2012). For an asymptotically constant reference, Faulwasser and Findeisen (2011) presented an MPC approach where convergence to zero of the tracking error is guaranteed.

4. SIMULATION RESULTS

MatLAB simulations were conducted to demonstrate the controller performance in tracking an acceleration command. Parameters used are typical in missile autopilot: \( T_s = 10\,\text{ms}, \alpha = 20^\circ, q = 1000\,\text{s}^{-1}, \delta = 45^\circ, \) and \( \bar{\delta} = 500\,\text{s}^{-1} \). For NMPC \( \epsilon_p = \epsilon_u = 10^{-4} \) and \( j_{\text{max}} = 10 \). Prediction horizon \( N \) and simulation time \( T \) are indicated for each individual result presented. Gurobi (Gurobi Optimization, 2013) was used to solve the QP pertaining to (7a–e). ACADO (Houska et al., 2011) was used to calculate \( \Phi(\cdot) \) using a 4th order Gauss-Legendre method in 16 steps, and the Jacobians in (11). In what follows, “rise time” is the time taken for the controlled missile to reach and stay within \( \pm 10\% \) of the desired acceleration command.

The presented NMPC scheme with an ellipsoidal terminal constraint is used to control a simulated missile model in tracking a given command. The missile model is initialised at the origin, and the controller is used to drive the model to track a 20, –5, 5 acceleration step for 0.5s each. The response exhibits zero steady-state error and no overshoot, with a rise time of about 150 ms (Fig. 2). This confirms stability in tracking set-point changes as posed in Proposition 1. The controller is also tested to track a non-constant reference, applying the method in Proposition 1 in choosing a terminal constraint at every sampling time. The controller achieves a lag of about 100 ms in tracking a 15, 2 Hz sinusoidal reference (Fig. 3). This result shows that the controller exhibits a bounded, although non-zero error, as per Remark 3.

To understand the quality of the presented NMPC as a controller on the basis of tracking performance and computational time, it is compared with the more typical LMPC approach outlined in Section 3.2. It is then benchmarked against the use of polytopic and no terminal regions.

4.1 Linear vs. nonlinear prediction model

Comparing NMPC and LMPC, both schemes achieve zero steady-state error and no overshoot for a step change
(Fig. 2), and a similar phase lag in tracking a sinusoidal reference (Fig. 3). In Fig. 2, NMPC exhibits an almost bang-bang response immediately after each step change, pushing the input u to its upper and lower limits for 0.1 s or so. This is less evident in LMPC, demonstrating that NMPC is more aggressive than LMPC, producing control commands with greater deviations from 0 (Table 1).

NMPC’s aggressiveness leads to faster rise times by about 10-20 ms (Table 2), and an overall better performance as measured by the cumulated difference between the actual and desired acceleration (Table 3). This highlights the capability of NMPC to push the missile nearer to its limits to better track a given command than LMPC.

The superior performance of NMPC for tracking comes at a cost of computational load. NMPC takes around 10-15 ms longer to solve in each sampling time (Table 4) than LMPC. Nonetheless, computational times achieved by NMPC have the same order of magnitude as a typical missile sampling time of 10 ms. This is promising for practical implementation, where a dedicated processor and compiled code can be used to reduce computational time.

Furthermore, in NMPC simulations, the iteration limit is set at \( j_{\text{max}} = 10 \) to let \( \Delta x < \epsilon_x \) and \( \Delta u < \epsilon_u \) in each sampling instant. For practical implementation, \( j_{\text{max}} \) can be set sufficiently small to guarantee that a solution is obtained within each sampling period. The first input \( u_1^* \) does not change significantly between iterations of Steps 1–3 in Algorithm 1. Changes mostly occur in the latter part of the sequence – which is not taken as feedback control – allowing \( j_{\text{max}} \sim 1 \) without much performance degradation.

As shown in Figs. 2 and 3, the performance with \( j_{\text{max}} = 1 \) is barely distinguishable from that with \( j_{\text{max}} = 10 \).

4.2 Ellipsoidal vs. polytopic vs. no terminal region

To further gauge the quality of the presented NMPC scheme, the use of an ellipsoidal terminal constraint (\( X_f = \mathcal{E} \)) is compared with a polytopic (\( X_f = \Pi \)) or no terminal constraint (no \( X_f \)). The choice of terminal constraint has an effect on the quality of the controller, both in tracking performance and solution time. In terms of tracking performance, the use of no \( X_f \) is shown to be able to achieve a performance similar to that with \( X_f = \mathcal{E} \) (Fig. 4), although there is a potential performance degradation (e.g. a steady-state error) when the terminal cost is poorly chosen.

With regards to solution time, although an appropriately chosen terminal constraint guarantees closed-loop stability, it inevitably has an associated computational burden, adding 1-5 ms of solution time for both NMPC and LMPC (Table 4). The constraint \( X_f = \Pi \) is linear and less computationally burdensome to solve than \( X_f = \mathcal{E} \) which quadratically constrains the pertaining optimisation problem. The results shown in Table 4 are consistent with this notion; NMPC and LMPC with \( X_f = \mathcal{E} \) take about 4 ms longer in solution time than those with \( X_f = \Pi \).

<table>
<thead>
<tr>
<th>( a_o )</th>
<th>( X_f = \mathcal{E} )</th>
<th>( X_f = \Pi )</th>
<th>No ( X_f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 5g )</td>
<td>1.00</td>
<td>0.86</td>
<td>1.00</td>
</tr>
<tr>
<td>( 10g )</td>
<td>1.00</td>
<td>0.69</td>
<td>1.00</td>
</tr>
<tr>
<td>( 15g )</td>
<td>1.00</td>
<td>0.64</td>
<td>1.00</td>
</tr>
<tr>
<td>( 20g )</td>
<td>1.00</td>
<td>0.66</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Values are normalised on results for NMPC with Ellipsoidal \( X_f \).

<table>
<thead>
<tr>
<th>( a_o )</th>
<th>( X_f = \mathcal{E} )</th>
<th>( X_f = \Pi )</th>
<th>No ( X_f )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 5g )</td>
<td>150</td>
<td>164</td>
<td>150</td>
</tr>
<tr>
<td>( 10g )</td>
<td>152</td>
<td>171</td>
<td>151</td>
</tr>
<tr>
<td>( 15g )</td>
<td>152</td>
<td>171</td>
<td>152</td>
</tr>
<tr>
<td>( 20g )</td>
<td>160</td>
<td>170</td>
<td>160</td>
</tr>
</tbody>
</table>

Values are normalised on results for NMPC with Ellipsoidal \( X_f \).

Table 1. \( \int_0^T u^2 dt \) in tracking a step change to various \( a_o \) from \( 0g \), \( T = 0.5 \) s, \( N = 25 \).

Table 2. Rise time in milliseconds, \( N = 25 \).

Table 3. \( \int_0^T (a - a_o)^2 dt \) in tracking a step change to various \( a_o \) from \( 0g \), \( T = 0.5 \) s, \( N = 25 \).

Table 4. Solution time in milliseconds, averaged for tracking constant and 2 Hz sinusoidal references with 5, 10, 15, 20 g steps and amplitudes respectively, \( T = 0.5 \) s, \( N = 25 \).

Average iterations taken in NMPC are given in brackets. Simulations were run on a 3.4GHz, 8.0GB, 64-bit desktop.
ACKNOWLEDGEMENTS

The authors would like to thank BAE Systems Australia for their support of this research.

REFERENCES


