I. INTRODUCTION

In this paper we establish the properties of a new control strategy called Reduced Parameterisation Model Predictive Control (RPMPC), applicable to linear, input constrained systems with real unstable modes. The strategy consists of two modes of operation: a recovery mode and a convergence mode. At each iteration of the recovery mode, the algorithm solves an optimisation problem over a dynamically computed horizon and with additional equality constraints on the input sequence. These constraints result from the substitution of the fully parameterised control input sequence (that is, where each control move is a parameter or decision variable for the optimisation) by a reduced parameterisation (RP) sequence, where a reduced number of independent control moves or parameters is used. The particular way of performing the RP substitution depends on the location of the current system’s state in relation to the facets of the feasible set for the optimisation, and is directly related to the structure of the null controllable set associated with the input constrained unstable system. When operating in convergence mode, on the other hand, the RPMPC algorithm solves a (small-sized) full parameterisation MPC over a user selected fixed horizon. By introducing recovery mode, the length of the overall MPC horizon and, hence, the closed-loop system region of attraction, can be expanded to a large size without the addition of a large number of decision variables.

The analysis presented in this paper establishes two important properties of the RPMPC algorithm. Firstly, we establish stability of the closed-loop system resulting from the application of RPMPC. Indeed, we show that the closed-loop system has a region of attraction that is an arbitrarily close approximation of the system’s null controllable set. Moreover, by using specific properties of controllable sets, we show that the algorithm evolves from recovery mode to convergence mode in a finite number of steps. Finally, asymptotic stability of the origin is established for the latter mode by the use of standard MPC stability arguments.

The second property of RPMPC that we analyse here is that of its overall computational complexity and its comparison with that of standard MPC. More specifically, we show that RPMPC exhibits a lower worst case complexity than MPC for long horizons or for systems with a high stable to unstable mode ratio. In addition, we present simulation results of average computational performance which (i) confirm the advantage of RPMPC over MPC predicted by the worst case analysis and (ii) show that the loss in performance, as measured by the MPC cost function, is modest.

In the companion paper [1], a more detailed account of the implementation is presented. The work presented here has been extended in [2] to systems with an arbitrary number of real unstable poles. For simplicity of exposition, we have chosen to develop here the case of two unstable modes.

II. SYSTEMS AND CONTROLLABLE SETS

We consider a class of single input, open-loop unstable, linear systems described by the following discrete-time state-space model,

\[
x_{k+1} = Ax_k + Bu_k,
\]

where \( x_k = \begin{bmatrix} x_{k|u}^T & x_{k|s}^T \end{bmatrix}^T \),

\[
A = \begin{bmatrix} A_u & 0 \\ 0 & A_s \end{bmatrix}, \quad B = \begin{bmatrix} B_u \\ B_s \end{bmatrix},
\]

and \( u_k \in U \triangleq [u_{\text{min}}, u_{\text{max}}] \).

The input saturation values satisfy \( u_{\text{min}} < 0 \) and \( u_{\text{max}} > 0 \), and \( (A_u, B_u) \) is controllable.

We will denote the null controllable set for (1)–(3) as \( \mathcal{R}^{\text{max}} \) which due to the systems’ structure can also be defined as the Cartesian product of an unstable \( \mathcal{R}^{\text{max}}_u \) and stable \( \mathcal{R}^{\text{max}}_s \) subsets. That is, \( \mathcal{R}^{\text{max}} \triangleq \mathcal{R}^{\text{max}}_u \times \mathcal{R}^{\text{max}}_s \). Further, [3,
Proposition 1) shows that $\mathcal{R}_{u}^{\text{max}}$ is unbounded and therefore the analysis, as presented here, only considers $\mathcal{R}_{u}^{\text{max}}$.

From [4], [5], [3], $\mathcal{R}_{u}^{\text{max}}$ is given by

$$
\mathcal{R}_{u}^{\text{max}} = \{ x_{0|u} \in \mathbb{R}^2 : \lim_{k \to \infty} \text{A}_{u} x_{k|u} + B_{u} u_{k} = 0 \}
$$

for $u_{k} \in \mathbb{U}$, $k = 0, 1, \ldots$.

(4)

The work in [2] shows that $\mathcal{R}_{u}^{\text{max}}$ can be arbitrarily closely approximated by an $N$-step controllable set, defined as

$$
\mathcal{R}^{N} = \{ x \in \mathbb{R}^2 : x = -\sum_{i=0}^{N-1} A_{u}^{i-1} B_{u} u_{i}, \quad \text{for} \quad u_{i} \in \mathbb{U}, \ i = 0, \ldots, N-1 \},
$$

via the selection of an arbitrarily large horizon $N$.

Detailed analysis of $N$-step controllable sets presented in [2] shows that they can be represented as a Minkowski sum (denoted by $\oplus$) of line segments called core generators.

Definition 1 (Core Generator): The $i$th core generator, is the line segment $g_i = \{ x \in \mathbb{R}^2 : x = \bar{g}_i u, \text{for}\ u \in \mathbb{U} \}$, for $i = 0, \ldots, N-1$, where $\mathbb{U}$ is as defined in (3) and the core generator direction $\bar{g}_i$ is given by $\bar{g}_i = -A_{u}^{i-1} B_{u}$.

Then, defining the set of core generators as $\mathcal{G} = \{ g_0, \ldots, g_{N-1} \}$, we have that $\mathcal{S}_{\ominus} (\mathcal{G}^{N}) = \mathcal{R}^{N}$.

An important property of $\mathcal{R}^{N}$ sets that will feature in proving the stability of RPMPC is the following:

Lemma 1 (Monotonic Inclusion of $\mathcal{R}^{N}$ Sets): For all horizons $N \geq 1$, the $N$-step controllable sets satisfy $\mathcal{R}^{N} \subset \mathcal{R}^{N+1}$.

Proof: See [2, Lemma 5.1].

The boundary of $\mathcal{R}^{N}$ is the set of extreme points that can be controlled to the origin in $N$ steps. The edges of this boundary can be described in the following way.

Definition 2 (Edge Sets): Let $\mathcal{R}^{N} \subset \mathbb{R}^2$ be such that $\mathcal{R}^{N} = \mathcal{S}_{\ominus} (\mathcal{G}^{N})$, where all generators in $\mathcal{G}^{N}$ are in general position. Then any edge $E_{i}^{N}$ of $\mathcal{R}^{N}$ has the structure

$$
E_{i}^{N} = \mathcal{S}_{\ominus} (g_{i}^{b,N}) \oplus z_{i}^{o,N},
$$

where $g_{i}^{b,N} \subset \mathcal{G}^{N}$ is termed the “edge basis” generator set and $z_{i}^{o,N} \in \mathbb{R}^2$ is the “edge offset”. Further we note that $g_{i}^{b,N}$ contains a single generator, with index denoted by $b$, that is, $g_{i}^{b,N} = \{ g_{b} \} \subset \mathcal{G}^{N}$.

Definition 3 (Core Edge): An edge whose basis generator is a core generator as per Definition 1 is a “core edge”.

We next introduce the concept of reduced parameterisation (RP) sequences. RP sequences are based on the structure of feasible control sequences for states originating on the edges of $\mathcal{R}^{N}$. It was shown in [2, Corollary 5.1] that, given an edge of $\mathcal{R}^{N}$ with basis generator index $b$, as per Definition 2, then a control sequence that can take any initial state on the edge to the origin in $N$ steps consists of all control moves $\{ u_0, \ldots, u_{N-1} \}$ having the same saturated value, and all control moves $\{ u_{N+1}, \ldots, u_{N-1} \}$ having the complementary saturated value. Based on this result, we introduce the following definition.

Definition 4 (RP Sequence): Let the edge $E_{i}^{N}$ of $\mathcal{R}^{N}$ be given as in Definition 2. Then the RP sequence associated with $E_{i}^{N}$ is given by the index set

$$
I_{k}^{RP} = \{ [0, b - 1], [b, b + 1, N - 1] \},
$$

where intervals $[i, j]$ having $i > j$ are removed from $I_{k}^{RP}$.

Note that the RP sequence of Definition 4 contains (at most) three sets of indices, $I_{k}^{RP} = [0, b - 1]$, $I_{k}^{RP} = [b, b + 1, N - 1]$, determined by the single edge basis index $b$.

Equivalently the application of each RP sequence index set $I_{k}^{RP}$ produces either a core generator as per Definition 1 or a composite generator as per the following definition.

Definition 5 (Composite Generator): Given a sequential index set $I_{k}^{RP} \subset \{ 0, \ldots, N - 1 \}$ such that $|I_{k}^{RP}| > 1$, the associated composite generator $g_{I_{k}^{RP}}^{RP}$ is the line segment,

$$
g_{I_{k}^{RP}}^{RP} = \{ x \in \mathbb{R}^2 : x = g_{I_{k}^{RP}}^{RP} u \text{ for } u \in \mathbb{U} \},
$$

where $g_{I_{k}^{RP}}^{RP} = \sum_{j \in I_{k}^{RP}} g_{j}^{RP}$.

In the following section we will describe the RPMPC algorithm, which employs the RP sequences of Definition 4.

III. RPMPC ALGORITHM

This section summarises the RPMPC algorithm presented in the companion paper [1]. In the subsequent sections we will refer to this summary for the analysis of feasibility, stability and complexity.

As in conventional MPC (see, for example [6]), RPMPC is based on solving a finite horizon optimal control problem at each sampling time, with the associated cost function

$$
V(X, U, N) = F(x_{N}) + \sum_{i=0}^{N-1} L(x_{i}, u_{i}).
$$

In (9), $X = \{ x_{0}, x_{1}, \ldots, x_{N} \}$, $U = \{ u_{0}, u_{1}, \ldots, u_{N-1} \}$, $N$ is the horizon length, $F(x_{N}) = \frac{1}{2} x_{N}^{T} P_{x_{N}}$, $P = P^{T} \geq 0$, represents the terminal cost and $L(x_{i}, u_{i}) = \frac{1}{2} (x_{i}^{T} Q_{x} x_{i} + u_{i}^{T} R_{u} u_{i})$, $Q = Q^{T} \geq 0$, $R = R^{T} > 0$, $i = 0, \ldots, N - 1$, is the per stage cost.

For consistency with the definition of the systems (1)–(2), we partition $Q$ and $P$ as follows:

$$
Q \triangleq \text{diag}(Q_{u}, Q_{s}), \quad P \triangleq \text{diag}(P_{u}, P_{s}).
$$

(10)

RPMPC departs from conventional MPC in that, at each iteration $k$ of the RPMPC algorithm, a new horizon $N = N_{k}$ is dynamically determined as $N_{k} = \max(N_{\text{min}}, N_{F})$, where $N_{\text{min}}$ is the recovery mode horizon whose computation is explained in Section III-A below and $N_{F}$ is a user defined fixed horizon employed in convergence mode.

The state will be constrained to the set $X = \{ \mathcal{R}^{N_{\text{max}}}, \mathcal{R}_{u}^{\text{max}} \}$, where $N_{\text{max}}$ has the form (5) and $N_{\text{max}}$ is a user-selected maximum horizon to approximate the null controllable set. In addition, equality constraints are introduced in the optimisation problem to reflect the fact that the $N$-move control sequence $U$ is substituted by an RP sequence $I_{k}^{RP}$ as per Definition 4. Note that each RP sequence $I_{k}^{RP}$ is associated with an edge of $\mathcal{R}^{N}$ and,

$^2$No two generators are collinear.

$^3$The subindex $k$ will become clear in Section III.
as discussed, its structure is determined by feasible control sequences for states originating on that edge. It was shown in [2, Theorem 6.1], that a feasible RP sequence for any state $x \in \mathcal{R}^N$ is that of the edge of $\mathcal{R}^N$ through which a ray emanating from the origin through $x$ exits $\mathcal{R}^N$. Thus, the determination of the appropriate RP sequence $\mathcal{T}_k^{RP}$ for the current state $x_k$ involves finding the corresponding “exit” edge of $\mathcal{R}^N$, as just described. A procedure to find the exiting edge is explained below in Section III-B.

Thus, the optimisation problem that RPMPC solves at each iteration for the current state $x_k = x$ is

$$\mathcal{P}_{N_k}(x) : V_{N_k}^{OPT}(x) \triangleq \min V(X, U, N),$$

subject to:

$$N = N_k,$$

$$x_{i+1} = Ax_i + Bu_i \quad \text{for} \quad i = 0, \ldots, N - 1, \quad x_0 = x,$$

$$u_i = u_i \quad \forall j, l \in \mathcal{T}_k^{RP} \quad \text{for} \quad i = 0, \ldots, |\mathcal{T}_k^{RP}|-1,$$

$$u_i \in U \quad \text{for} \quad i = 0, \ldots, N - 1,$$

$$x_i \in X \quad \text{for} \quad i = 0, \ldots, N, \quad x_N \in X_f \subset X,$$

where $U$ is as in (3), $X_f = \{x_N \in \mathbb{R}^n : x_N[u] = 0\}$ is the terminal set and $\mathcal{T}_k^{RP}$ is the RP sequence as given in Definition 4. Each set of indices $\mathcal{T}_k^{RP} \subset \mathcal{T}_k^{RP}$, for $i = 0, \ldots, |\mathcal{T}_k^{RP}|-1$ introduces a sequence of equality constraints to the optimisation problem. Further for the case of $N_k = N_F$ (convergence mode), $\mathcal{T}_k^{RP} = \emptyset$, or equivalently all control input equality constraints are removed.

The solution to the optimisation problem (11)-(16) results in the optimal control sequence,

$$U_k^{OPT} \triangleq \{u_0^{OPT}, u_1^{OPT}, \ldots, u_{N_k-1}^{OPT}\}. \quad (17)$$

Then the control applied by RPMPC at time $k$ is the first element of (17). That is, we set $u_k = u_0^{OPT}$.

In the following subsections we describe the algorithms used by RPMPC to obtain $N_{min}$ and the RP set $\mathcal{T}_k^{RP}$.

A. Determination of $N_{min}$

The recovery horizon $N_{min}$ is defined as

$$N_{min} = \min \left\{ N \in [N_F, N_{max}] : x_k \in \mathcal{R}^N \right\}, \quad (18)$$

with $\mathcal{R}^N$ of the form (5). The determination of $N_{min}$ is implemented by combining a binary search algorithm and a set membership determination procedure explained in Section III-B below. The binary search is as follows.

Algorithm 1 (Search for $N_{min}$): Produce a sequence of test horizons $N^t$ using a binary search over the range $[N_F, N_{max}] \subset \mathbb{N}$ and test for $x_k \in \mathcal{R}^N$ until $N_{min} = N^t$ such that $x_k \in \mathcal{R}^{N_{min}}$ and $x_k \notin \mathcal{R}^{N_{min}-1}$.

B. Set Membership and Reduced Parameterisation

This subsection describes both a technique for obtaining an RP set $\mathcal{T}_k^{RP}$ used in (14) for the control sequence and the elements required to determine if $x_k \in \mathcal{R}^{N^t}$ (Algorithm 1).

1) Selecting a Reduced Parameterisation: As discussed above, selecting an appropriate RP sequence for the current state $x_k$ involves finding the “exit” edge of $\mathcal{R}^N$. This is implemented in the following algorithm which is based on constructing a series of test polytopes (whose precise structure is given in [1] Definitions 10 and 12), each designed to contain a portion of the ray connecting the origin with $x_k$, until the last polytope in the series contains the exiting edge.

Algorithm 2 (Membership Determination): Given a test state $x^t$ and a test horizon $N^t$ (Algorithm 1), determine the edge of $\mathcal{R}^{N^t}$ through which a ray $r^t$ from the origin through $x^t$ exits $\mathcal{R}^{N^t}$ and if $x^t \in \mathcal{R}^{N^t}$.

1) Set $k = 0$, $\mathcal{T}_k^{RP} = \{0, [1, N - 1]\}$ and construct the initial test polytope $\mathcal{R}_k^{OP}$

2) Determine the edge $\mathcal{E}_k^{OP}[m_k]$ of $\mathcal{R}_k^{OP}$ through which the test ray $r^t$ exits.

3) If the exit edge $\mathcal{E}_k^{OP}[m_k]$ is also an edge of $\mathcal{R}^{N^t}$ then test for $x^t \in \mathcal{R}^{N^t}$ and terminate the algorithm.

4) If the exit edge $\mathcal{E}_k^{OP}[m_k]$ is not an edge of $\mathcal{R}^{N^t}$, then increment the iteration $k = k + 1$ and calculate a new test polytope $\mathcal{R}_{k+1}$.

5) Return to step 2. o

The initial test polytope is chosen as $\mathcal{R}_0 = \{x_0 \in \mathcal{R}^{N^t} : g_{0}^{RP} \cup g_{0}^{RP} \}$ where $g_{0}^{RP} = g_0$ is the first core generator as per Definition 1 and $g_{0}^{RP}$ is a composite generator resulting from the index set $\mathcal{T}_0^{RP} = \{1, \ldots, N - 1\}$ as per Definition 5.

To construct a new test polytope around $\mathcal{E}_k^{OP}[m_k]$ we employ a technique called an edge expansion.

Algorithm 3 (Edge Expansion): For a composite edge $\mathcal{E}_k^{OP}[m_k]$ (see [1] Definition 10) of the test polytope $\mathcal{R}_k^{OP}$ (see [1] Definition 12), the edge expansion proceeds as follows,

1) Initialise the following new sets $G_k^{RP} = G_k^{OP}$, $G_k^{RP} = G_k^{OP}$ and $G_{k+1}^{RP} = G_{k+1}^{OP}$.

2) Select the composite generator $g_{k+1}^{RP} \in G_{k+1}^{RP}$.

3) Re-index the elements of $G_{k+1}^{RP}$, $G_{k+1}^{RP}$ and $G_{k+1}^{RP}$ with respect to the composite generator index $c_{k+1}$.

4) Partition $g_{k+1}^{RP}$ and replace it in $G_{k+1}^{RP}$ with its resulting partitioned generator pair.

5) Derive a new test polytope offset $z_{k+1}^{RP}$ from the sets $(G_{k+1}^{RP}, U_{k+1}^{RP})$ as per [1] Definition 12.

6) Construct $\mathcal{R}_{k+1}^{OP}$ using $G_{k+1}^{RP}$ and $z_{k+1}^{RP}$.

In step 4, partitioning a composite generator involves dividing the interval of indices that describes the set of core generator directions which form the composite generator direction, into two balanced sub-intervals of core generator directions as explained in the definition below.

Definition 6 (Composite Generator Partitioning): Given a composite generator $g_{c_k}^{RP}$ defined as

$$g_{c_{k+1}}^{RP} = \{x \in \mathbb{R}^2 : x = z_{c_{k+1}}^{RP} + u_{c_{k+1}}^{RP}, \quad u_{c_{k+1}}^{RP} \in U\},$$

where $z_{c_{k+1}}^{RP} = \sum_{i=s(c_{k+1})}^{e(c_{k+1})} c_{k+1}$ and $c_{k+1}$ is a core generator direction as in Definition 1 for $i = s(c_{k+1}), \ldots, e(c_{k+1})$.

Then partitioning $g_{c_{k+1}}^{RP}$ results in the two new generators $g_{c_{k+1}}^{RP}$ and $g_{c_{k+1}}^{RP}$ where $u_{c_{k+1}}^{RP} \in U$, and
Algorithm (Algorithm 2) identifies the edge of the test polytope resulting in a time varying horizon and hence a dynamic cost function for recovery mode. The dual modes of RPMPC result in a time varying horizon and hence a dynamic cost function for recovery mode. The following theorem uses the properties of test polytopes that guarantees this property for any test polytope resulting in the search process. An important property in establishing the stability proof.

**Theorem 1 (Search Exits an Edge of \( R_N \))**: Given an initial state \( x^i \in R_N \), then the membership determination algorithm (Algorithm 2) identifies the edge of \( R_N \), through which a ray emanating from the origin through \( x^i \) exits, in a finite number of iterations.

**Proof**: See [2, Theorem 6.2].

IV. PROPERTIES OF ALGORITHM 2

This section uses the properties of \( N \)-step controllable sets, RP sequences and test polytopes to establish the validity of Algorithm 2.

The test polytopes are a key element of the algorithms search process. An important property in establishing the validity of the search is that each test polytope must contain a validity of the search must be that each test polytope must contain a new test polytope offset generator set \( G^{t}_{[k+1]} \), its saturated values set \( U^{t}_{[k+1]} \) and [1] Definition 12, we construct in step 5 the new test polytope offset vector \( z_{[k]} \).

It is of note that due to the initialisation, \( z_{[k]} \) is equivalently constructed using the previous iteration’s offset vector \( z_{[k]} \) and test edge offset \( z_{m[k]} \), where \( z_{m[k]} \) is as per [1] Definition 10.

**Lemma 2 (Test Polytope Covers Expanded Edge)**: Performing an edge expansion (Algorithm 3) on a composite edge \( E_{m[k]} \) which is the \( m \)th edge of a test polytope \( R_{[k]} \), results in a new test polytope \( R_{[k]} \) with edges \( E_{m[k]} \), such that \( E_{m[k]} \subset R_{[k]} \) and \( E_{m[k]} \notin E_{[k]} \).

**Proof**: See [2, Lemma 6.2].

The following theorem uses the properties of test polytopes to show that Algorithm 2 terminates in a finite number of iterations and identifies an exit edge of \( R_N \).

**Theorem 1 (Search Exits an Edge of \( R_N \))**: Given an initial state \( x^i \in R_N \), then the membership determination algorithm (Algorithm 2) identifies the edge of \( R_N \), through which a ray emanating from the origin through \( x^i \) exits, in a finite number of iterations.

**Proof**: See [2, Theorem 6.2].

V. STABILITY

Stability for standard MPC is established via a Lyapunov argument using the cost function of the optimisation problem as a Lyapunov function candidate. As indicated by (12), the dual modes of RPMPC result in a time varying horizon and hence a dynamic cost function for recovery mode. The combination of a dynamic cost function and the effect of RP makes it difficult to use RPMPC’s cost function as a Lyapunov function candidate. However, we will show that, for recovery mode, a far simpler approach to stability involves using a combination of properties of the RPMPC algorithm and of \( N \)-step controllable sets.

Below we analyse the stability properties of both recovery and convergence modes of RPMPC and show that in concert these properties can be used to establish asymptotic stability of the origin for all feasible initial states. We first require a preliminary result.

**Proposition 1 (Horizon Contraction for RPMPC)**: Let \( x_0 \in R_{N_{F}} \setminus R_{N_{F}} \) be a given initial state for the system (1)–(3). Let the control sequence \( u_* = \{u_0^*, \ldots, u_{N-1}^*\} \) be the result of the RPMPC algorithm for \( x_0 \). Then the subsequent state \( x_1 = Ax_0 + Bu_0^* \) satisfies \( x_1 \in R_{N-1} \).

**Proof**: See [2, Proposition 6.1].

The next result establishes the stability properties of RPMPC recovery mode.

**Lemma 3 (Stability of RPMPC Recovery Mode)**: For a given initial state \( x_k \in R_{N_{F}} \times R_{N_{F}} \) where \( N_k > N_{F} \) then the application of RPMPC to system (1)–(3) results in \( x_{k+N_{k}-N_{F}} \in R_{N_{F}} \times R_{N_{F}} \).

**Proof**: See [2, Lemma 6.5].

If the current state’s unstable sub-state \( x_{k_{iW}} \) is such that \( x_{k_{iW}} \in R_{N_{F}} \) then the RPMPC algorithm executes convergence mode. Once in convergence mode RPMPC is standard 10 steps with a horizon length of \( N_{F} \) therefore the following stability argument can be used.

**Lemma 4 (Stability of RPMPC Convergence Mode)**: Let \( x_k \in R_{N_{F}} \times R_{N_{F}} \) be the current state of the system (1)–(3). Assume that \( A \) has no eigenvalue with modulus equal to one\(^3\). Then the cost function from matrices in (10) can be selected such that by applying the RPMPC algorithm to system (1)–(3) the resulting state trajectory asymptotically converges to the origin.

**Proof**: See [2, Lemma 6.6].

**Theorem 2 (Asymptotic Stability of RPMPC)**: Let an approximation to the null controllable set be given by \( \overline{R}_{N_{F}} \setminus R_{N_{F}} \) and let an initial state be given by \( x_k \in \overline{R}_{N_{F}}. \) Then system (1)–(3) in closed-loop with the RPMPC algorithm with particular selections of the cost function matrices in (10) has an asymptotically stable equilibrium point at the origin.

**Proof**: See [2, Theorem 6.3].

VI. WORST CASE COMPLEXITY

This section analyses the worst case complexity of the RPMPC algorithm to show that it provides a significant advantage over standard MPC for long control horizons.

To compare the complexity of RPMPC and MPC we require a common unit of computation. Complexity of standard MPC using some of the more advanced interior-point methods is quoted to be \( O(N(n+m)^3) \) where \( N \) is the horizon length of the optimisation problem, \( n \) is the number of system states and \( m \) is the number of system inputs [7], [8].

We note that this figure is stated as the order of complexity or equivalently the dominant contribution to complexity. Hence, we describe this dominant computation as a mathematical operation and provide the following definition.

---

\(^3\)This assumption has been added to simplify the presentation of the stability proof.
Proceedings of the European Control Conference 2009 • Budapest, Hungary, August 23–26, 2009

MoB4.2

Definition 7 (MPC Solution Complexity): The complexity of solving an MPC problem for a system with \( n \) states and 1 input over a horizon of \( N \) is

\[
c_{MPC}(N) = N (n + 1)^3
\]  

(21)

mathematical operations.

For RPMPC, the computational complexity of recovery mode can be attributed to two distinct operations: the search for \( N_{\text{ms}} \) and the solution of an RP optimisation problem. Since the complexity of MPC is derived from other references [7], the focus of this section is in analysing the search for \( N_{\text{ms}} \). The search for \( N_{\text{ms}} \), consists of producing a sequence of test horizons \( N^f_{[k]} = \{N^f_0[k], \ldots, N^f_{t-1}[k]\} \) and testing the membership of \( x_k \) with respect to each \( N \)-step controllable set corresponding to \( N^f_{[k]} \). Each set membership test consists of a sequence of edge support line constructions, test ray intersection parameter determinations and some set manipulation operations. Thus, for RPMPC the equivalent dominant computational complexity is achieved by two matrix multiplications, as presented in the following definition.

Definition 8 (RPMPC Test Line Complexity): The complexity of constructing a test edge support line and intersection parameter is \( c_{\text{te}} = 2 \) mathematical operations.

In summary, by combining the number of test horizons, the number of test polytope edge support lines and their construction complexity, gives the overall complexity of the search for \( N_{\text{ms}} \). These components are presented in the following subsections.

A. Producing the Test Horizon Sequence

The sequence of test horizons is produced by Algorithm 1 which as noted is a binary search algorithm. A well known result for a binary search is that for a search space consisting of \( d \) elements, the worst case number of searches is given by \( \log_2(d + 1) \). Hence, in the specific case of the recovery mode of the RPMPC algorithm, where \( d = N_{\text{max}} - N_F \) elements, the worst case number of test horizons, denoted by \( f \), is

\[
f = \log_2(N_{\text{max}} - N_F) + 1.
\]  

(22)

Since the complexity of a membership determination depends on the horizon length the following definition gives the worst case sequence of test horizons.

Definition 9 (Worst Case Test Horizon Sequence): For the worst case where the solution horizon is \( N_{\text{max}} \) the sequence of test horizons is defined as

\[
N^f_{[k]} = \{N^f_0[k], N^f_1[k], \ldots, N^f_{t-2}[k], N_{\text{max}}\},
\]  

(23)

\( \text{where } N^f_0[k] = (N_{\text{max}} - N_F)/2 + N_F, N^f_1[k] = (N_{\text{max}} - N^f_0[k])/2 + N_0, \ldots, N^f_{t-2}[k] = (N_{\text{max}} - N^f_{t-3}[k])/2 + N_{t-3}, \text{ and } N_{\text{max}} \text{ is the user defined fixed horizon length (i.e., the lower bound of recovery mode) and } N_{\text{max}} \text{ is the user defined maximum horizon length (see Section III).} \) \footnote{Assume, for simplicity, that } \( N_{\text{max}} \) and \( N_F \) are even numbers.

B. Performing a Membership Determination

The membership determination is implemented by Algorithm 2. This algorithm results in a sequence of 2-dimensional test polytopes being constructed for each membership determination. After each test polytope’s support lines are constructed their intersection with the test ray is determined so that an exit edge can be identified. Therefore, to determine the complexity of a membership determination we first establish the worst case number of test polytopes that are constructed for a selected horizon length.

Lemma 5 (Worst Case Number of Test Polytopes): For a test horizon \( N^f \), the worst case number of test polytopes generated by Algorithm 2 is given by \( C_{tp}(N^f) \) as follows,

\[
C_{tp}(N^f) = \log_2(N^f - 1) + 2.
\]  

(24)

\textbf{Proof:} See [2, Lemma 6.7 equation (6.97)].

C. Overall Complexity

Using the previously determined components we are now in a position to describe the overall worst case complexity of the RPMPC algorithm.

Lemma 6 (Complexity of Membership Search): The worst case complexity of the membership search, from RPMPC’s recovery mode, for iteration \( k \) is given by

\[
T_{\text{ms}} = \sum_{i=0}^{f-1} 2^i C_{tp}(N^f_{[k]}).
\]  

(25)

\textbf{Proof:} See [2, Lemma 6.8].

Adding the complexity of recovery mode and convergence mode results in the following theorem.

Theorem 3 (Complexity of RPMPC): The worst case complexity of the RPMPC algorithm at iteration \( k \) is

\[
T_{\text{RPMPC}} = \begin{cases} 
    c_{MPC}(3) + T_{\text{ms}} & \text{if } N_{\text{ms}} > N_F, \\
    c_{MPC}(N_F) & \text{if } N_{\text{ms}} \leq N_F,
\end{cases}
\]  

(26)

where \( T_{\text{ms}} \) is as defined in \( (25) \) and \( c_{MPC}(\cdot) \) is the complexity of solving an MPC problem as per Definition 7.

\textbf{Proof:} See [2, Theorem 6.4].

Since there is a clear advantage to RPMPC operating in convergence mode compared to MPC, provided \( N_{\text{max}} > N_F \), we do not investigate this aspect of complexity any further. Hence, the following illustrations and subsequent analysis only considers RPMPC operating in recovery mode.

To illustrate the comparative worst case complexity of RPMPC and MPC Fig. 1 shows plots of the number of mathematical operations versus horizon length, for the expressions \( (26) \) and \( (21) \).

From Fig. 1 we see that due to the polylogarithmic dependency of RPMPC’s complexity and the linear dependency of MPC’s complexity on the horizon length, there will always be a value of the latter at which the two complexity functions intersect. As seen from the plots, if the relative number of unstable modes is small compared with the order of the
system, MPC’s cubic complexity with respect to system order causes the horizon for equal complexity to be relatively small. Thus, for systems with a large stable to unstable mode ratio, RPMPC exhibits a lower complexity than MPC.

VII. AVERAGE PERFORMANCE

Worst case analysis, although a useful indicator, may not accurately reflect how an algorithm will perform in practice. This section provides an illustration of the practical performance of RPMPC versus MPC. As an example of the relative computational advantage of the RPMPC over MPC for long horizons Fig. 2(left) shows the resulting execution times of RPMPC versus MPC for a grid of initial states that cover the $N$-step controllable sets for a system with two unstable modes. These states were chosen such that RPMPC operates in recovery mode. The different symbols represent execution times produced by each implementation for the maximum horizons $N_{max}$ of 30, 50 and 70. Further, to ensure an equitable comparison both implementations were terminated after the same number of iterations. The execution of the RPMPC algorithm, is terminated when $N_{min} = N_F$. This same number of iterations is used for the MPC algorithm.

It can be seen from Fig. 2(left) that since this is a representation of average computation, the times are dependent on the horizon and the initial state selected. But overall the RPMPC execution times are lower than for its equivalent MPC implementation. This result is consistent with the worst case complexity analysis presented in Section VI. Lastly Fig. 2(right) illustrates the overall closed-loop cost $\sum_{k=0}^{N} x_k^T Q x_k + u_k^T R u_k$ for a number of different initial states $x_0$ and where, for both MPC and RPMPC, $N$ was chosen such that $x_N$ belongs to a small neighbourhood of the origin. From this figure we see that there are cases where the MPC cost is significantly smaller, but in most cases the change in cost is modest.

VIII. CONCLUSION

This paper has presented stability, and computational complexity properties of a novel MPC algorithm for a class of input constrained $n^{th}$ order systems with two real unstable poles. The new algorithm, Reduced Parameterisation MPC (RPMPC), is summarised in Section III and detailed in [1].

Among the analysed properties of RPMPC, we have established that the RPMPC algorithm must converge in a finite number of iterations and that under mild assumptions closed-loop stability can be guaranteed.

Both rigorous worst case analysis and average computational requirement analysis showed distinct advantages for RPMPC. The comparative analysis concluded that for long horizons or for systems with a high stable to unstable mode ratio, RPMPC performed better than MPC.

REFERENCES