Model Predictive Control with Reach-avoid Analysis

Dejin Ren1,2, Wanli Lu3, Jidong Lv3, Lijun Zhang1,2 and Bai Xue1,2∗
1State Key Lab. of Computer Science, Institute of Software, CAS, Beijing, China
2University of Chinese Academy of Sciences, Beijing, China
3National Engineering Research Center of Rail Transportation Operation and Control System, Beijing Jiaotong University, Beijing, China
{rendj, zhanglj, xuebai}@ios.ac.cn, {luwanli, jdlv}@bjtu.edu.cn

Abstract

In this paper we investigate the optimal controller synthesis problem, so that the system under the controller can reach a specified target set while satisfying given constraints. Existing model predictive control (MPC) methods learn from a set of discrete states visited by previous (sub-)optimized trajectories and thus result in computationally expensive mixed-integer nonlinear optimization. In this paper a novel MPC method is proposed based on reach-avoid analysis to solve the controller synthesis problem iteratively. The reach-avoid analysis is concerned with computing a reach-avoid set which is a set of initial states such that the system can reach the target set successfully. It not only provides terminal constraints, which ensure feasibility of MPC, but also expands discrete states in existing methods into a continuous set (i.e., reach-avoid sets) and thus leads to nonlinear optimization which is more computationally tractable online due to the absence of integer variables. Finally, we evaluate the proposed method and make comparisons with state-of-the-art ones based on several examples.

1 Introduction

Control synthesis is a fundamental problem which automatically constructs a control strategy that induces a system to exhibit a desired behavior. Due to the ability of handling control and state constraints and yielding high performance control systems [Camacho and Alba, 2013], control design methods based on MPC have gained great popularity and found wide acceptance in industrial applications, ranging from autonomous driving [Verschueren et al., 2014; Kabzan et al., 2019] to large scale interconnected power systems [Ernst et al., 2008; Mohamed et al., 2011].

In MPC design methods one issue is to guarantee feasibility of the successive optimization problems [Scokaert and Rawlings, 1999]. Because MPC is ‘greedy’ in its nature, i.e., it only searches for the optimal strategy within a finite horizon, an MPC controller may steer the state to a region starting from where the violation of hard state constraints cannot be avoided. Although this feasibility issue could be solved by using a sufficiently long prediction horizon, we may not be able to afford the computational overhead due to the limited computational resources. Consequently, several solutions towards the feasibility issue are proposed [Zheng and Morari, 1995; Zeng et al., 2021; Ma et al., 2021]. Besides the feasibility issue of satisfying all hard state constraints [Zheng and Morari, 1995; Zeng et al., 2021; Ma et al., 2021], a system is also desired to achieve certain performance objective in an optimal sense. In existing literature, stability performance of approaching an equilibrium within the MPC framework is extensively studied. One common solution of achieving stability is adding Lyapunov functions as terminal costs and/or corresponding invariant sets as terminal sets [Michalska and Mayne, 1993; Limon et al., 2005; Limón et al., 2006; Mhaskar et al., 2006; de la Peña and Christofides, 2008; Wu et al., 2019; Grandia et al., 2020]. This has motivated significant research work on applications of this control design to nonlinear processes [Yao and Shekhar, 2021]. However, in many real applications stability performance is demanding. For example, a spacecraft rendezvous may require the chaser vehicle to be at a certain position relative to the target, moving towards it with a certain velocity. All of these quantities are specified with some tolerance, forming a target region in the state space. When the chaser enters that region a physical connection would be made and the maneuver is complete. However, since the target velocity is non-zero, the region is not invariant and stability cannot be achieved. Regarding this practical issue, the formulation in this paper replaces the notion of stability with reachability: given a target set, the system will achieve the reachability objective of reaching the target set in finite time successfully. To the best of our knowledge, the learning model predictive control (LMPC) proposed in [Rosolia and Borrrelli, 2017], which utilizes the historical data to improve suboptimal controllers iteratively, is the only method within the MPC framework which can solve this problem. However, it leads to mixed-integer nonlinear programming problems which are fundamentally challenging to solve.

In this work, we consider control tasks where the goal is to steer the system from a starting configuration to a target set in finite time, while satisfying state and input constraints. The control task is formalized as a reach-avoid optimization problem.
Consider a discrete-time dynamical system \( x(t + 1) = f(x(t), u(t)), t \in \mathbb{N}, \) where \( f(\cdot) : \mathbb{R}^n \times \mathbb{R}^m \rightarrow \mathbb{R}^n \) is the system dynamics, \( x(0) \in \mathbb{R}^n \) is the state and \( u(\cdot) : \mathbb{N} \rightarrow \mathcal{U} \subseteq \mathbb{R}^m \) is the control policy. 

### Problem Statement

A control policy is a sequence of control inputs \( \pi = \{u(i)\}_{i \in \mathbb{N}} \), where \( u(i) \in \mathcal{U} \). Furthermore, we define \( \Pi \) as the set of all control policies.

Given a safe set \( \mathcal{X} = \{x \in \mathbb{R}^n \mid \omega(x) \leq 0\} \) and a target set \( \mathcal{T} = \{x \in \mathbb{R}^n \mid g(x) \leq 0\} \) with \( \mathcal{T} \subseteq \mathcal{X} \), a control policy \( \pi \in \Pi \) is safe with respect to a state \( x_0 \), if the control policy \( \pi \) will drive system (1) starting from \( x(0) = x_0 \) to reach the target set \( \mathcal{T} \) in finite time while staying inside the safe set \( \mathcal{X} \) before the first target hitting time. The associated cost of reaching the target set \( \mathcal{T} \) safely is defined below.

#### Definition 2

Given a state \( x(0) = x_0 \in \mathcal{X} \setminus \mathcal{T} \), and a safe policy \( \pi = \{u(i)\}_{i \in \mathbb{N}} \in \Pi \), the cost with respect to the state \( x_0 \) and the policy \( \pi \) is defined below:

\[
Q(x_0, \pi) = \sum_{k=0}^{L-1} h(x(k), u(k)),
\]

where \( h(x(k), u(k)) \) is the cost function, and \( \mathcal{T} \) is the target set.

In this paper, we would like to synthesize a safe control policy \( \pi \in \Pi \) with respect to a specified initial state \( x_0 \) such that system (1) will enter the target set \( \mathcal{T} \) with the minimal cost in finite time while staying inside the safe set \( \mathcal{X} \) before the first target hitting time.

### Problem 1

We attempt to solve the following reach-avoid optimization problem:

\[
J(x_0) = \min_{T, u(i), i=0, \ldots, T-1} \sum_{i=0}^{T-1} h(x(i), u(i)),
\]

subject to

\[
\begin{align*}
&x(i + 1) = f(x(i), u(i)), x(0) = x_0, \\
&u(i) \in \mathcal{U}, x(i) \in \mathcal{X}, \\
&T \in \mathbb{N},
\end{align*}
\]

where \( T \) is the first time of hitting the target set \( \mathcal{T} \).

Due to uncertainties on the target hitting time, it is challenging to solve optimization (4) directly. However, the computational complexity can be reduced when searching for a feasible sub-optimal solution. In this paper we will adopt a learning strategy to solving (4), which iteratively improves upon already known suboptimal policies as the LMPC algorithm in [Rosolia and Borrelli, 2017]. Consequently, an initial feasible policy is needed, as formulated in Assumption 1.

#### Assumption 1

Assume an initial policy \( \pi^{0} = \{u^{0}(i)\}_{i \in \mathbb{N}} \), which can drive system (1) starting from \( x_0 \) to the target set \( \mathcal{T} \) in finite time safely, is available. The corresponding trajectory of system (1) can be obtained and is denoted by \( x^{0}(i) \), where

\[
\begin{align*}
x^{0}(i + 1) &= f(x^{0}(i), u^{0}(i)), \quad i = 0, \ldots, L^0 - 1, \\
x^{0}(0) &= x_0, \quad x^{0}(L^0) \in \mathcal{T}.
\end{align*}
\]

#### Remark 1

The availability of a feasible control policy is not restrictive in practice for a number of applications. For instance, with race cars one can always run a path at very low speed to obtain a control policy.

### 2.2 Guidance-barrier Functions

In this subsection we introduce guidance-barrier functions. They not only provide terminal constraints in our MPC method escorting system (1) to the target set \( \mathcal{T} \) safely, but also generate a set which curves out a viability space for system (1) to reach the target set \( \mathcal{T} \) safely.
Definition 3. Given the safe set $X$, target set $T$ and a factor $\lambda \in (1, \infty)$, a bounded function $v(x) : Y \to \mathbb{R}$ is a guidance-barrier function of system (1) with the feedback controller $\hat{u}(\cdot) : X \to U$, if it satisfies the following constraints:

$$
\begin{align*}
&v(f(x, \hat{u}(x))) \geq \lambda v(x), \forall x \in X \setminus T, \\
&v(x) \leq 0, \forall x \in Y \setminus X, \\
&v(x) \leq M, \forall x \in T, \\
&v(x_0) > 0,
\end{align*}
$$

(5)

where $Y = \{ y \mid y = f(x, u), u \in U, x \in X \} \cup X$, and $M$ is a user-defined positive number.

When $f(x, \hat{u}(x))$ is polynomial over $x$, and $X$ is semi-algebraic set, i.e., $f(x, \hat{u}(x)), w(x) \in \mathbb{R}[x]$, a set $\mathcal{Y}$ of the form $\{ x \mid w_0(x) \leq 0 \}$ with $w_0(x) \in \mathbb{R}[x]$ can be obtained using program (3) in [Zhao et al., 2022].

Remark 2. It is worth remarking here that if (5) holds, for system (1) with $\hat{u}(\cdot) : X \to U$, the induced trajectory starting from $x_0$ will hit the target set $T$ within a bounded amount of time being less or equal to $\log_{\lambda} M (\text{It can obtained according to } T^T v(x_0) \leq v(x(T)) \leq M, \text{ where } T \text{ is the first hitting time of } T.)$

A reach-avoid set $R = \{ x \in X \mid v(x) > 0 \}$ can be computed via solving constraint (5), which is a set of states such that there exists a control policy $\pi \in \Pi$ driving system (1) to enter the target set $T$ in finite time while staying inside the safe set $X$ before the first target hitting time.

Theorem 1. Given the safe set $X$, target set $T$ and a factor $\lambda \in (1, \infty)$, if $v(x) : Y \to \mathbb{R}$ is a guidance-barrier function of system (1) with the feedback controller $\hat{u}(\cdot) : X \to U$, then $R = \{ x \in X \mid v(x) > 0 \}$ is a reach-avoid set.

Theorem 1 can be assured by Corollary 1 in [Zhao et al., 2022]. Due to space limitations, we omitted the proof herein.

Remark 3. One of admissible control policies $\pi \in \Pi$ such that system (1) satisfies the reach-avoid specification can be constructed by the feedback controller $\hat{u}(\cdot) : X \to U$: when system (1) is in state $x(i) \in R$, the corresponding control action is $u(i) = \hat{u}(x(i))$. We denote such a control policy by $\pi_{\hat{u}}$ in the rest of this paper.

3 Reach-avoid Model Predictive Control

In this section we elucidate our learning-based algorithm for solving optimization (4) in Problem 1, which is built upon a so-called reach-avoid model predictive control (RAMPC). The proposed RAMPC is constructed based on a guidance-barrier function.

Our RAMPC algorithm is iterative and at each iteration it mainly consists of three steps. The first step is to synthesize a feedback controller by interpolating the suboptimal state-control pair obtained in the previous iteration. Then, a guidance-barrier function satisfying (5) with the synthesized feedback controller is computed. Finally, based on the computed guidance-barrier function a MPC controller, together with its resulting state trajectory, is generated online. The framework of the algorithm is summarized in Alg. 1.

For solving (5) in Alg. 1, when $f(x, \hat{u}(x))$ is polynomial over $x$, and $Y, X$ and $T$ are semi-algebraic sets, i.e.,

\begin{align*}
\text{Algorithm 1} & \quad \text{The framework for solving optimization (4).} \\
& \text{Require: system (1); initial state } x_0; \text{ safe set } X; \text{ target set } T; \text{ control input set } U; \text{ feasible state-control trajectory } \{(x^0(i), u^0(i))\}_{i=0}^{L_0-1}, \text{ of which the cost is } J^0(x_0) = \sum_{i=0}^{L_0-1} h(x^0(i), u^0(i)); \text{ factor } \lambda \text{ and bound } M \text{ in (5);} \text{ maximum iteration number } K; \text{ prediction horizon } N \text{ in RAMPC; termination threshold } \xi > 0. \\
& \text{Ensure: Return } J^*(x_0). \\
& \text{for } j = 0 : K \text{ do} \\
& \quad 1 \quad \text{apply interpolation techniques to compute a feedback controller } \hat{u}(\cdot) : X \to U \text{ based on the } j\text{-th state-control trajectory } \{(x^j(i), u^j(i))\}_{i=0}^{L_j-1}; \\
& \quad 2 \quad \text{compute } v^j(x) \text{ via solving (5) with } \hat{u}(x) = \hat{u}^j(x); \\
& \quad 3 \quad \text{solve RAMPC optimization, which is constructed with } v^j(x), \text{ to obtain a state-control pair } \\
& \quad \quad \text{of which the cost } J^{j+1}(x_0) \text{ is } \lambda \text{-Lyapunov condition } \text{[Liu et al., 2013]} \text{ for robust convex optimization can be employed to solve constraint (5).} \text{ The basis of these approaches is the Almost Lyapunov condition } \text{[Liu et al., 2020]}, \text{ which allow the Lyapunov conditions to be violated in restricted subsets of the space while still ensuring stability properties. Although results from these approaches lack rigorous guarantees, nice empirical performances demonstrated the practicality of these approaches in existing literature (e.g., [Chang and Gao, 2021]). We will revisit it in our future work.} \\
& \quad \text{if } J^{j+1}(x_0) - J^j(x_0) \geq -\xi \text{ then} \\
& \quad \quad \text{Return } J^*(x_0) = J^{j+1}(x_0); \\
& \quad \end{align*}

end if
be the optimal solution to (7) at time $t$
region denotes a target set.
RAMPC method and discrete states in the LMPC method, the green
In (7), the terminal constraint
\begin{equation}
\begin{aligned}
v^{j-1}(x_{N|t}) \geq & \begin{cases}
\lambda^N v^{j-1}(x_0), & \text{if } t = 0, \\
\lambda^{j-1}(x_{N|(t-1)}), & \text{otherwise.}
\end{cases}
\end{aligned}
\end{equation}
guarantees that the terminal state $x_{N|t}$ lies in the reach-avoid
set $\mathcal{R}^{j-1} = \{x \in \mathcal{X} \mid v^{j-1}(x) > 0\}$, which carves out a
larger continuous viability space (visualized as the cyan re-
region in Fig. 1) for system (1) to be capable of achieving the
reach-avoid objective. This is different from the LMCP
method in [Rosolia and Borrelli, 2017], which restricts ter-
ninal states within previously explored discrete states (vis-
ualized as pink points in Fig. 1) and thus leads to computa-
tionally demanding mixed-integer nonlinear optimization. As to
the practical computations of the cost $Q^j(\cdot, \pi_{\hat{u}_{j-1}}) : \mathcal{R}^{j-1} \to \mathbb{R}$, we will give a detailed introduction in Subsection 3.2.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{An illustration of continuous (reach-avoid) sets in our
RAMPC method and discrete states in the LMPC method, the green
region denotes a target set.}
\end{figure}

Let
\begin{equation}
\begin{aligned}
\mathbf{u}^{*,j}_{0:N|t} = \{u^{*,j}_{i|t}\}_{i=0}^{N-1} \quad \text{and} \quad x^{*,j}_{0:N|t} = \{x^{*,j}_{i|t}\}_{i=0}^{N-1}
\end{aligned}
\end{equation}
be the optimal solution to (7) at time $t$ and $J_t^{\text{RAMPC,j}}(x^j_t)$
the corresponding optimal cost. Then, at time $t$ of the iteration
$j$, the first element $u^j(t) = u^*_{0|t}$ of $\mathbf{u}^{*,j}_{0:N|t}$ is applied to the
system (1) and thus the state of system (1) turns into
\begin{equation}
\begin{aligned}
x^{j}_{t+1} = x^{*,j}_{t|t} = f(x^j(t), u^j(t)),
\end{aligned}
\end{equation}
where $x^j(t) = x^{*,j}_{0|t}$, which will be the used to update the ini-
tial state in (7) for subsequent computations. Once there exist
time $t \in \mathbb{N}$ and $t \leq N$ s.t. $x^j_{t|t} \in \mathcal{T}$, we terminate computations in
this iteration: from the state $x^j_{0|t}$ on, we will apply the control
actions $\{u^{*,j}_{i|t}\}_{i=1}^{N-1}$ successively to system (1).

Let $\pi_t^j = \{u^j(t)\}_{0 \leq i \leq N-1}$ be the control policy applied to
system (1) by solving optimization (7). The resulting
state-control pair $\{x^j(i), u^j(i)\}_{0 \leq i \leq L_j-1}$, where $x^j(L_j) \in \mathcal{T}$,
is obtained. We can conclude that the performance of $\pi^j$ is no worse than that of $\pi^{j-1}$, i.e., $J^{j-1}(x_0) = \sum_{i=0}^{L_j-1} h(x^j(i), u^j(i)) \geq \sum_{i=0}^{L_j-1} h(x^j(i), u^j(i)) = J^{j}(x_0)$. This conclusion can be justified by Theorem 3.

Under Assumption 2, in Theorem 2 we show that the
RAMPC (7) is feasible and system (1) with controllers computed
by solving (7) satisfies the reach-avoid property, and in
Theorem 3 show that the $j$-th iteration cost $J^j(x_0)$ is non-
increasing as $j$ increases in RAMPC (7). Their proofs can be
found in the appendix of the extended version [Ren et al.,
2023].

**Assumption 2.** At each iteration $0 \leq j \leq K$ the computed
reach-avoid set $\mathcal{R}^j = \{x \in \mathcal{X} \mid v^j(x) > 0\}$ via solving (5) is
non-empty. In addition, we assume that it includes the state
trajectory $\{x^j(i)\}_{i=0}^{L_j-1}$.

**Theorem 2.** For each iteration $j \leq K$, the RAMPC (7) is
feasible for $1 \leq t \leq L_j - 1$; also, system (1) with controllers obtained by solving RAMPC (7) can reach the target
set $\mathcal{T}$ within the time interval $[0, \log_{\lambda_{\max}}(1/\delta)]$ while satisfying all of state and input constraints.

**Theorem 3.** Consider system (1) with controllers obtained by
solving (7). Then, the iteration cost $J^j(x_0)$ is non-increasing
with the iteration index $j$.

### 3.2 Estimating Terminal Costs via Scenario Optimization

In practice, the exact terminal cost $Q^j(\cdot, \pi_{\hat{u}_{j-1}}) : \mathcal{R}^{j-1} \to \mathbb{R}$
is challenging, even impossible to obtain. In this subsection,
we present a linear programming method based on the sce-
nario optimization [Calafiore and Campi, 2006] to compute an
approximation of the terminal cost $Q^j(\cdot, \pi_{\hat{u}_{j-1}}) : \mathcal{R}^{j-1} \to \mathbb{R}$
in the probably approximately correct (PAC) sense.

Let $\{x_t\}_{t=1}^{N_x}$ be the independent samples extracted from
$\mathcal{R}^{j-1} = \{x \in \mathcal{X} \mid v^{j-1}(x) > 0\}$ according to the uniform
distribution $\mathbb{P}$, and $Q^j(x_t, \pi_{\hat{u}_{j-1}})$ be the corresponding
value of the roll-out from $x_t$ with the controller $\pi_{\hat{u}_{j-1}}$. Such
cost can be simply computed by summing up the cost along
the closed-loop realized trajectory until the target set $\mathcal{T}$ is
reached. Finally, using the set of sample states and corre-
spending costs $\{(x_t, Q^j(x_t, \pi_{\hat{u}_{j-1}}))\}_{t=1}^{N_x}$, we approximate the
terminal cost $Q^j(\cdot, \pi_{\hat{u}_{j-1}}) : \mathcal{R}^{j-1} \to \mathbb{R}$ using the scenario
optimization [Calafiore and Campi, 2006].

A linearly parameterized model template $Q^j_{\alpha}(c_1, \ldots, c_t, x)$, $k \geq 1$ is utilized, which is a linear
function in unknown parameters $c = (c_1, \ldots, c_t)$ but can be nonlinear over $x$. Then we construct the following
linear program over $c$ based on the family of given datum

---

1This assumption can be realized by interpolating the feedback controller $\hat{u}^j(\cdot) : \mathcal{X} \to \mathcal{U}$ s.t. $\hat{u}^j(x^j(t)) = u^j(i)$, $i = 0, \ldots, L_j - 1$. This requirement mainly serves for our theoretical analysis since it can guarantee that our algorithm can iteratively improve the policy (Theorem 3). However, in practical this requirement is not indis-

---

5440
In this section we evaluate our RAMPC algorithm, i.e., Alg. solving optimization (4) in Algorithm 2.

Denote the optimal solution to (11) by \((e^*, \delta^*)\). The discrepancy between \(Q^*_a(e^*, x)\) and \(Q^a(x, \pi_{a,j-1})\) is formally characterized by two parameters: the error probability \(\epsilon \in (0, 1)\) and confidence level \(\beta \in (0, 1)\).

**Theorem 4.** Let \((e^*, \delta^*)\) be an optimal solution to (11), \(\epsilon \in (0, 1)\), \(\beta \in (0, 1)\) and

\[
\epsilon \geq \frac{2}{N^t}(\ln \frac{1}{\beta} + t + 1).
\]

Then we have that with at least \(1 - \beta\) confidence,

\[
\mathbb{P}(\{x \in \mathcal{R} \mid |Q^*_a(e^*, x) - Q^a(x, \pi_{a,j-1})| \leq \delta^*\}) \geq 1 - \epsilon.
\]

Thus, we relax optimization (4) into the following form:

\[
x^*_a_{t+1} = \underset{u_{k|t}}{\text{min}} \{ \sum_{k=0}^{N-1} h(x_{k|t}, u_{k|t}) + Q^a_{j-1}(e^*, x_{N|t}) \}
\]

s.t. \(x_{k+1|t} = f(x_{k|t}, u_{k|t}), u_{k|t} \in U, x_{k|t} \in X, k = 0, 1, \ldots, N - 1, v(x_{N|t}) \geq \begin{cases} \lambda^N v(x_0), & \text{if } t = 0, \\ \lambda v(x_{N|t-1}), & \text{otherwise}. \end{cases} \)

where \(Q^a_{j-1}(e^*, x_{N|t})\) is the approximate terminal cost at the state \(x_{N|t}\), which is obtained via solving (11).

**Proposition 1.** With at least \(1 - \beta\) confidence,

\[
|J^*_a_{t+1}(x^*_t) - J^*_a_{t+1}(x^*_t)| \leq \delta^*
\]

holds with the probability larger than or equal to 1 - \(\epsilon\), i.e.,

\[
\mathbb{P}(\{x^*_t \in \mathcal{R}^{-1} \mid |J^*_a_{t+1}(x^*_t) - J^*_a_{t+1}(x^*_t)| \leq \delta^*\}) \geq 1 - \epsilon.
\]

Relying on (12), we summarize our RAMPC algorithm for solving optimization (4) in Algorithm 2.

**Algorithm 2** The RAMPC algorithm for solving (4).

**Require:** system (1) with an initial state \(x_0\), a safe set \(\mathcal{X}\), a target set \(\mathcal{T}\) a control input set \(\mathcal{U}\) and a feasible state-control trajectory \(\{(x^0(i), u^0(i))\}_{i=0}^{L-1}\), of which the corresponding cost is \(J^0(x_0) = \sum_{i=0}^{L-1} h(x^0(i), u^0(i))\); factor \(\lambda\) and bound \(M\) in (5); iteration number \(K\), prediction horizon \(N\) and termination threshold \(\xi > 0\); probability error \(\epsilon\) and confidence level \(\delta\) in PAC approximations.

**Ensure:** Return \(J^+(x_0)\). for all \(j = 0 : K\) do

1. apply interpolation techniques to compute \(\hat{u}^j(\cdot) : \mathcal{X} \rightarrow \mathcal{U}\) for the j-th state-control trajectory \(\{(x^j(i), u^j(i))\}_{i=0}^{L-1}\);
2. obtain \(\mathcal{R}^j = \{x \in \mathcal{X} \mid v^j(x) > 0\}\) via solving (5) with \(\hat{u}(x) = \hat{u}^j(x)\);
3. compute the PAC terminal cost \(J^a_j(\cdot) : \mathcal{R}^j \rightarrow \mathbb{R}\) via solving optimization (11);
4. Solving MPC optimization (12) with \(v^j\) to obtain a state-control pair \(\{(x^{j+1}(i), u^{j+1}(i))\}_{i=0}^{L-1}\) and the cost \(J^{j+1}(x_0)\);

if \(|J^{j+1}(x_0) - J^j(x_0)| \leq \xi\) then 

Return \(J^+(x_0) = J^{j+1}(x_0)\);
end if 
end for

[Rosolia and Borrelli, 2017] are solved using YALMIP [Lofberg, 2004]. In addition, we take \(\hat{u}^j(\cdot)\) as a linear function to interpolate the j-th state-control trajectory \(\{(x^j(i), u^j(i))\}_{i=0}^{L-1}\) at each iteration \(j \geq 1\). The configuration parameters in Alg. 2 for all examples are shown in Table 1.

<table>
<thead>
<tr>
<th>Example</th>
<th>(\lambda)</th>
<th>(M)</th>
<th>(N)</th>
<th>(K)</th>
<th>(\xi)</th>
<th>(\epsilon)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ex. 1</td>
<td>1.001</td>
<td>1</td>
<td>4</td>
<td>8</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>Ex. 2</td>
<td>1.001</td>
<td>1</td>
<td>3</td>
<td>8</td>
<td>0.1</td>
<td>0.05</td>
</tr>
<tr>
<td>Ex. 3</td>
<td>1.001</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>0.002</td>
<td>0.1</td>
</tr>
</tbody>
</table>

**Table 1:** Configuration parameters in Alg. 2 for examples.

**Example 1.** Consider the drone system from [Rosolia and Borrelli, 2017],

\[
x_{t+1} = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} x_t + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_t
\]

where \(x_t = (p_t, v_t)^T\), \(p_t\) and \(v_t\) are respectively the position and velocity of the drone at time \(t\), \(u_t\) is the control input and \(dt\) is the control interval which is equal to 0.1.

We assume \(\mathcal{X} = \{(p, v)^T \mid \frac{p^2}{2} + \frac{v^2}{2} - 1 \leq 0\}\), initial state \(x_0 = (4, -6)^T\), target set \(\mathcal{T} = \{(p, v)^T \mid p^2 + v^2 - 0.5^2 \leq 0\}\) and control input set \(\mathcal{U} = \{u \mid -0.5 \leq u \leq 0.5\}\). The set \(\mathcal{Y} = \{(p, v)^T \mid \frac{p^2}{2} + \frac{v^2}{2} - 2 \leq 0\}\) in constraint (5) is obtained by solving a semi-define program as in [Zhao et al., 2022]. The cost in optimization (4) is \(h(x, u) = \|x\|^2 + \|u\|^2\). For simplicity of presentation herein, we do not present the initial state-control trajectory which is a long sequence. The initial state-control trajectory is induced by the feedback controller \(\hat{u}^0(p, v) = -0.04p - 0.1v\). For scenario optimization (11),
Consider the Euler version with the time step $\Delta t = 0.05$ of the controlled reversed-time Van der Pol oscillator in [Drazin and Drazin, 1992]:

$$
\begin{align*}
  x_1(t+1) &= x_1(t) - \Delta tx_2(t) \\
  x_2(t+1) &= x_2(t) - \Delta t(1 - x_1^2(t))x_2(t) - x_1(t) + u(t),
\end{align*}
$$

with the safe set $X = \{(x_1, x_2)^T \mid \left(\frac{x_1}{2}\right)^2 + \left(\frac{x_2}{2}\right)^2 \leq 1\}$.

Initial state $x_0 = (1.2, 1)^T$, target set $T = \{(x_1, x_2)^T \mid x_1^2 + x_2^2 - 0.2^2 \leq 0\}$ and input set $U = \{u \mid -0.5 \leq u \leq 0.5\}$.

In (5), the set $Y = \{(x_1, x_2)^T \mid \left(\frac{x_1}{2}\right)^2 + \left(\frac{x_2}{2}\right)^2 - 2 \leq 0\}$ is computed by solving a semi-definite program in [Zhao et al., 2022]. In addition, the cost $h(x, u) = \|x\|_2^2 + \|u\|_2^2$, where $x = (x_1, x_2)^T$, is adopted. The initial trajectory is induced by the controller $\hat{u}(x) \equiv 0$. For scenario optimization (11), we use a template of the polynomial form $Q_0(x, p, v) = c_1 + c_2x_1 + c_3x_2 + c_4x_1x_2 + c_5x_1^2 + c_6x_2^2$ and $N' = 207$ sampled points which can be computed via Theorem 4. LMPC uses the same prediction horizon and termination conditions (i.e., $N$ and $\xi$).

Some trajectories generated by our RAMPC algorithm and the LMPC algorithm are visualized in Figure 3. Our RAMPC algorithm terminates after the third iteration, but the LMPC one does not terminate after eight iterations. Figure 3 shows that the initial trajectory and the trajectory generated by the first iteration in the LMPC algorithm are too close to be indistinguishable within the first few time steps, which on the other hand further reflects that the controller generated in the first iteration of the LMPC algorithm may not improve the performance induced by the initial control policy. Besides, it is observed that the eighth trajectory from the LMPC algorithm looks not stable and has strong oscillations, which are not expected in practice. In contrast, the trajectory generated by our algorithm is smoother.

Table 3 summarizes the iteration costs and the computation times at each iteration of our RAMPC algorithm and the LMPC one. The iteration cost induced by the initial trajectory is 64.3087, after three iterations our RAMPC algorithm reduces the cost to 29.2824 with the computation time of 31.2841s. In contrast, the LMPC algorithm needs more than 8 iterations with the computation time of almost more than 3 hours to achieve the same cost. This striking contrast definitely demonstrates that our RAMPC algorithm can solve optimization (4) more efficiently for some cases.

**Example 2.** Consider the Euler version with the time step $\Delta t = 0.1$ of the controlled 3D Van der Pol oscillator from

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Iteration Cost</th>
<th>Time Cost(seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAMPC</td>
<td>LMPC</td>
<td>RAMPC</td>
</tr>
<tr>
<td>0</td>
<td>369.8267</td>
<td>369.8267</td>
</tr>
<tr>
<td>1</td>
<td>215.1007</td>
<td>222.2482</td>
</tr>
<tr>
<td>2</td>
<td>215.1002</td>
<td>217.3604</td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>215.3008</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>215.1003</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>215.1044</td>
</tr>
<tr>
<td>total time</td>
<td>6.3304</td>
<td>8.2259</td>
</tr>
</tbody>
</table>

Table 2: Iteration cost and computation time for Example 1.
In this paper we propose a novel RAMPC algorithm for solving a reach-avoid optimization problem. The RAMPC algorithm is built upon MPC and reach-avoid analysis. Rather than solving computationally intractable mixed-integer non-linear optimization online, it addresses computationally more tractable nonlinear optimization. Moreover, due to the incorporation of reach-avoid analysis, which expands the viability space, the RAMPC algorithm can provide better controllers with a smaller number of iterations. Several numerical examples demonstrated the advantages of our RAMPC algorithm.

5 Conclusion
In this paper we proposed a novel RAMPC algorithm for solving a reach-avoid optimization problem. The RAMPC algorithm is built upon MPC and reach-avoid analysis. Rather than solving computationally intractable mixed-integer non-linear optimization online, it addresses computationally more tractable nonlinear optimization. Moreover, due to the incorporation of reach-avoid analysis, which expands the viability space, the RAMPC algorithm can provide better controllers with a smaller number of iterations. Several numerical examples demonstrated the advantages of our RAMPC algorithm.
Acknowledgments

This research was supported by the NSFC under grant No. 61836005, CAS Project for Young Scientists in Basic Research under grant No. YSBR-040, NSFC under grant No. 52272329, and CAS Pioneer Hundred Talents Program.

References


[Verschueren et al., 2014] Robin Verschueren, Stijn De Bruyne, Mario Zanon, Janick V Frasch, and Moritz


