总结

本文提出了一种用于**跟踪受约束非线性系统 piecewise 参考指令** 的 MPC，该控制器能够处理片断恒定设定点信号的**突然大幅变化**。

该控制器将人工参考作为额外的决策变量，并使预测轨迹与人工参考的跟踪误差以及该参考与所提供设定点之间的距离最小化。

稳定设计基于终端成本函数和终端约束区域的适当选择。实践证明，使用终端约束和不使用终端约束都能实现稳定。

假定参考值在足够长的时间内保持不变，如果是容许（admissible，可实现）的，控制器会渐近地将系统导向参考值。否则，受控系统将收敛到容许的稳定状态，使成本函数最小化。

> 该控制器将人工参考作为额外的决策变量，并使预测轨迹与人工参考的跟踪误差以及该参考与所提供设定点之间的距离最小化。

这里人工参考作为额外的决策变量的作用即为参考管理（reference governor），保证的是递归可行性。稳定性的保证还是通过终端约束和终端成本。

系统模型

模型：

$$\begin{aligned} x^+ &= f(x,u) \\ y &= h(x,u) \end{aligned} \tag{1}$$

约束：

$$(x(k),u(k)) \in \mathcal{Z} \tag{2}$$

稳态：

$$x_s = f(x_s,u_s) \tag{3a}$$
$$y_s = h(x_s,u_s). \tag{3b}$$

文章为了防止约束“**激活**”（active），建立了受限（restricted）约束集：

这里的激活我理解为数值达到约束边界。

$$\hat{\mathcal{Z}} = \{z : z+e \in \mathcal{Z}, \forall |e| \leq \varepsilon\} \tag{4}$$

ε 是一个非常小的正数。

那么约束未激活的稳态集合可以表示为：

$$\mathcal{Z}_s = \{(x,u) \in \hat{\mathcal{Z}} : x = f(x,u)\}, \tag{5}$$
$$\mathcal{Y}_s = \{y = h(x,u) : (x,u) \in \mathcal{Z}_s\}. \tag{6}$$

那么集合 Z_s 内的稳态就可以逼近约束边界但不激活约束。

假设1： 稳态可以由目标输出 y_s 唯一决定，且连续变化：

$$x_s = g_x(y_s),\ u_s = g_u(y_s). \tag{7}$$

针对这个非线性受约束系统，要设计一个 MPC ： 对于给定的参考点（设定点或输出目标）y_t，闭环系统是稳定的，能**满足约束条件**，并收敛（尽可能接近）到由给定设定点 y_t 计算得到的平衡点。此外，即使设定点 y_t 突然变为一个不同恒定值，也要**满足闭环稳定性**。

MPC 设计

$$\begin{aligned} V_{N_c,N_p}(x,y_t;\mathbf{u},y_s) = &\sum_{j=0}^{N_c-1} \ell(x(j)-x_s, u(j)-u_s) \\ &+\sum_{j=N_c}^{N_p-1} \ell(x(j)-x_s, \kappa(x(j),y_s)-u_s) \\ &+V_f(x(N_p)-x_s, y_s)+V_O(y_s-y_t) \end{aligned} \tag{8}$$

前三项用于衡量跟踪误差，最后一项衡量参考指令的偏差。

结合约束、不变集，写出优化问题：

$$\min_{\mathbf{u},y_s} \quad V_{N_c,N_p}(x,y_t;\mathbf{u},y_s) \tag{9a}$$
$$s.t.$$
$$x(0) = x, \tag{9b}$$
$$x(j+1) = f(x(j),u(j)), \quad j=0,\cdots,N_c-1 \tag{9c}$$
$$(x(j),u(j)) \in \mathcal{Z}, \quad j=0,\cdots,N_c-1 \tag{9d}$$
$$x(j+1) = f(x(j),\kappa(x(j),y_s)),\ j=N_c,\cdots,N_p-1 \tag{9e}$$
$$(x(j),\kappa(x(j),y_s)) \in \mathcal{Z}, \quad j=N_c,\cdots,N_p-1 \tag{9f}$$
$$x_s = g_x(y_s),\ u_s = g_u(y_s), \tag{9g}$$
$$(x(N_p),y_s) \in \Gamma. \tag{9h}$$

稳态的对应计算公式 (9 g) 如果计算不出来，也可以直接替换为：

Notice that constraints (9g) could be replaced by

$$\begin{aligned} x_s &= f(x_s, u_s) \\ y_s &= h(x_s, u_s) \end{aligned}$$

being x_s and u_s new decision variables. This set of constraints are suitable when the functions $g_x(\cdot)$ and $g_u(\cdot)$ are not explicitly known.

不变集的计算

In order to derive the stability conditions, it is convenient to extend the notion of invariant set for tracking introduced in [21], [16] to the nonlinear case, which is defined as follows:

Definition 1 (Invariant set for tracking): For a given set of constraints $\mathcal{Z}$, a set of feasible setpoints $\mathcal{Y}_t \subseteq \mathcal{Y}_s$ and a local control law $u = \kappa(x, y_s)$, a set $\Gamma \subset \mathbb{R}^n \times \mathbb{R}^p$ is an (admissible) invariant set for tracking for system (1) if for all $(x, y_s) \in \Gamma$ we have that $(x, \kappa(x, y_s)) \in \mathcal{Z}$, $y_s \in \mathcal{Y}_t$, and $(f(x, \kappa(x, y_s)), y_s) \in \Gamma$. ■

This set can be read as the set of initial states and setpoints (x, y_s) that provide an admissible evolution of the system (1) controlled by the control law $u = \kappa(x, y_s)$ with a constant reference y_s.

线性系统

根据不变集的定义：

$$\mathcal{O}^w_{\infty,\lambda} = \{w : A^i_w w \in W_\lambda, \forall i \geq 0\}$$

其中 w 是状态， W_λ 是约束。

这个集合是通过迭代的形式定义的，就可以用迭代来求解：

步骤1：定义初始集合

初始集合通常定义为满足状态约束的所有状态：

$\mathcal{X}_0 = \mathcal{X}$

这表示初始集合 $\mathcal{X}_0$ 包含所有满足状态约束的状态。

步骤2：计算系统的 Predecessor Set

对于每一个时刻 k，计算系统在当前状态集合 $\mathcal{X}_k$ 下，通过控制输入 $\mathcal{U}$ 所能达到的所有状态。这个集合称为 $\mathcal{X}_{k+1}$ 的 Predecessor Set。

$\mathrm{Pre}(\mathcal{X}_k) = \{x \mid Ax + Bu \in \mathcal{X}_k \text{ for some } u \in \mathcal{U}\}$

步骤3：计算新的状态集合

新的状态集合 $\mathcal{X}_{k+1}$ 是初始状态集合 $\mathcal{X}$ 和 Predecessor Set $\mathrm{Pre}(\mathcal{X}_k)$ 的交集：

$\mathcal{X}_{k+1} = \mathcal{X} \cap \mathrm{Pre}(\mathcal{X}_k)$

步骤4：迭代计算

重复步骤2和步骤3，逐步更新状态集合，直到集合不再变化（即达到收敛），得到最大正不变集。

1. **初始化**：设定初始状态集合 $\mathcal{X}_0 = \mathcal{X}$。
2. **迭代更新**：

$$\mathcal{X}_{k+1} = \mathcal{X} \cap \mathrm{Pre}(\mathcal{X}_k)$$

   其中 $\mathrm{Pre}(\mathcal{X}_k)$ 通过计算 x 满足 $Ax + Bu \in \mathcal{X}_k$ 来获得。

3. **检查收敛性**：如果 $\mathcal{X}_{k+1} = \mathcal{X}_k$，则认为集合已经收敛，即找到了最大正不变集。

步骤5：验证结果

验证最终的集合 $\mathcal{X}_\infty$ 是否满足正不变性，即对于所有 $x \in \mathcal{X}_\infty$，存在控制输入 $u \in \mathcal{U}$ 使得 $Ax + Bu \in \mathcal{X}_\infty$。

也可以通过 Lyapunov 函数求解不变集。

使用求解得到的矩阵 P 定义不变集：

$\mathcal{X}_f = \{x \mid V(x) \leq c\}$

其中 $V(x) = x^T P x$， c 是一个常数，使得集合 $\mathcal{X}_f$ 包含所有满足Lyapunov函数减小条件的状态。

非线性系统

非线性系统无法通过求解线性优化问题离线得到不变集，一般采用Lyapunov方法：

方法1：Lyapunov方法

Lyapunov方法是分析非线性系统稳定性和不变集的经典方法。步骤如下：

1. **选择Lyapunov函数**：
   - 选择一个适当的Lyapunov函数 $V(x)$，通常是一个正定函数，满足 $V(x) > 0$ 当 $x \neq 0$，且 $V(0) = 0$。
2. **验证Lyapunov条件**：
   - 确定Lyapunov函数的导数 $\dot{V}(x)$ 沿系统轨迹 $x_{k+1} = f(x_k, u_k)$ 的变化情况。
   - 如果存在一个集合 $\mathcal{X}$，使得在该集合内 $\dot{V}(x) < 0$，则该集合内的状态会趋向于稳定。
3. **确定不变集**：
   - 通过Lyapunov函数 $V(x)$ 的水平集来确定不变集：
     $\mathcal{X}_V = \{x \mid V(x) \leq c\}$
   - 其中常数 c 是根据系统具体情况确定的，使得在集合 $\mathcal{X}_V$ 内，Lyapunov函数 $V(x)$ 不增加。

这种方法要在非线性模型中找到一个 Lyapunov 函数，比较困难；文章：

> Chen H, Allgöwer F. A quasi-infinite horizon nonlinear model predictive control scheme with guaranteed stability[J]. Automatica, 1998, 34(10): 1205-1217.

提供了一种线性化的方法，因为线性模型的 Lyapunov 方程比较好找：

where $A := (\partial \mathbf{f}/\partial \mathbf{x})(\mathbf{0}, \mathbf{0})$ and $B := (\partial \mathbf{f}/\partial \mathbf{u})(\mathbf{0}, \mathbf{0})$. If equation (8) is stabilizable, then a linear state feedback $\mathbf{u} = K\mathbf{x}$ can be determined such that $A_K := A + BK$ is asymptotically stable. Consequently, we can state the following lemma.

Lemma 1. Suppose that the Jacobian linearization of the system (1) at the origin is stabilizable. Then,

(a) the following Lyapunov equation

$$(A_K + \kappa I)^{\mathrm{T}} P + P(A_K + \kappa I) = -Q^* \qquad (9)$$

admits a unique positive-definite and symmetric solution P, where $Q^* = Q + K^{\mathrm{T}} R K \in \mathbb{R}^{n \times n}$ is positive definite and symmetric; $\kappa \in [0, \infty)$ satisfies

$$\kappa < -\lambda_{\max}(A_K). \qquad (10)$$

(b) There exists a constant $\alpha \in (0, \infty)$ specifying a neighborhood Ω_α of the origin in the form of

$$\Omega_\alpha := \{\mathbf{x} \in \mathbb{R}^n | \mathbf{x}^{\mathrm{T}} P \mathbf{x} \leq \alpha\} \qquad (11)$$

such that

(i) $K\mathbf{x} \in U$, for all $\mathbf{x} \in \Omega_\alpha$, i.e., the linear feedback controller respects the input constraints in Ω_α,

(ii) Ω_α is invariant for the nonlinear system (1) controlled by the local linear feedback $\mathbf{u} = K\mathbf{x}$,

(iii) for any $\mathbf{x}_1 \in \Omega_\alpha$, the infinite horizon cost

$$J^\infty(\mathbf{x}_1, \mathbf{u}) = \int_{t_1}^{\infty} (\|\mathbf{x}(t)\|_Q^2 + \|\mathbf{u}(t)\|_R^2)\, \mathrm{d}t$$

subject to nonlinear system (1), starting from $\mathbf{x}(t_1) = \mathbf{x}_1$ and controlled by the local linear state feedback $\mathbf{u} = K\mathbf{x}$, is bounded from above as follows:

$$J^\infty(\mathbf{x}_1, \mathbf{u}) \leq \mathbf{x}_1^{\mathrm{T}} P x_1. \qquad (12)$$

V_O 的设计

文章没有提及，根据观察文中仿真的设定，V_O 设置的比较大，数值上一般在 Q 矩阵的 100 倍。

The MPC for tracking without terminal constraint has been tested on the plant. The horizons used are $N_c = 5$ and $N_p = 20$. The stage cost function used is $l(x, u) = |x - x_s|_Q^2 + |u - u_s|_R^2$ with $Q = I_4$ and $R = 0.01I_2$ as weighting matrices. The function $V_O = \alpha|y_s - y_t|_\infty$, with $\alpha = 100$ has been chosen

V_f 设计

文章未说明如何设计，在仿真部分，文章直接使用了线性化后的系统方程，求解Lyapunov方程得到 cost to go：

as offset cost function. The cost-to-go is taken of the form $V_f(x - x_s, y_s) = (x - x_s)'P(x - x_s)$. The gain matrix K for terminal control law and the matrix P for the cost-to-go have been calculating by solving an LMI problem, and are given by:

$$K = \begin{bmatrix} -0.0282 & 0.0916 & 0.1384 & -0.4479 \\ 0.0883 & 0.0439 & -0.4553 & 0.4704 \end{bmatrix}$$

$$P = \begin{bmatrix} 10.9654 & 0.6257 & 9.6267 & 1.3360 \\ 0.6257 & 10.2893 & 0.7929 & 8.8616 \\ 9.6267 & 0.7929 & 42.2449 & 5.5129 \\ 1.3360 & 8.8616 & 5.5129 & 38.0525 \end{bmatrix}$$

文章提供的解决方法

终端相等约束

直接令终端状态等于稳态，这样就不需要终端代价和终端约束。这种方法会损失递归可行性。

$$\min_{\mathbf{u},y_s} \quad V_N(x,y_t;\mathbf{u},y_s) \tag{10a}$$
$$s.t.$$
$$x(0)=x, \tag{10b}$$
$$x(j+1)=f(x(j),u(j)), j=0,\cdots,N-1 \tag{10c}$$
$$(x(j),u(j))\in\mathcal{Z}, \quad j=0,\cdots,N-1 \tag{10d}$$
直接相等 $$x_s=g_x(y_s),\ u_s=g_u(y_s) \tag{10e}$$
$$y_s\in\mathcal{Y}_t \tag{10f}$$
$$x(N)=x_s. \tag{10g}$$

The MPC for tracking based on the equality constraint is very simple to design and ensures closed-loop stability. However, it has also some drawbacks with respect to the general case: (i) the domain of attraction is potentially smaller, that is, $X_{N_c} \subseteq X_{N_c,N_p}$ and (ii) the benefits of using a prediction horizon larger than the control horizon in terms of closed-loop performance [24] may be lost. In the following section it is presented a method to design a stabilizing MPC for tracking with $N_p > N_c$ such that the terminal constraint is removed from the optimization problem.

无终端约束设计

V_f 终端代价是 Lyapunov 函数，参考上述不变集的计算，可以用 V_f 来表达不变集：

$$\Gamma_\alpha=\{(x,y_s): V_f(x-g_x(y_s),y_s)\le\alpha\} \tag{11}$$

such that Γ_α is an invariant set for tracking. Notice that there exists a constant $\alpha>0$ satisfying this condition since for all reachable setpoint in $\mathcal{Y}_t$, the constraints are not active thanks to the condition (4).

V_f 围绕 0 点递减，只要选择一个合适的 α 满足系统约束即可。

对于优化问题（9）：

$$
\begin{aligned}
&\min_{\mathbf{u},y_s} \quad V_{N_c,N_p}(x,y_t;\mathbf{u},y_s) && \text{(9a)}\\
&s.t.\\
&x(0)=x, && \text{(9b)}\\
&x(j+1)=f(x(j),u(j)),\quad j=0,\cdots,N_c-1 && \text{(9c)}\\
&(x(j),u(j))\in\mathcal{Z},\quad j=0,\cdots,N_c-1 && \text{(9d)}\\
&x(j+1)=f(x(j),\kappa(x(j),y_s)),\ j=N_c,\cdots,N_p-1 && \text{(9e)}\\
&(x(j),\kappa(x(j),y_s))\in\mathcal{Z},\quad j=N_c,\cdots,N_p-1 && \text{(9f)}\\
&x_s=g_x(y_s),\ u_s=g_u(y_s), && \text{(9g)}\\
&(x(N_p),y_s)\in\Gamma. && \text{(9h)}
\end{aligned}
$$

其优化目标函数为：

$$
\begin{aligned}
V_{N_c,N_p}(x,y_t;\mathbf{u},y_s)=&\sum_{j=0}^{N_c-1}\ell(x(j)-x_s,u(j)-u_s)\\
&+\sum_{j=N_c}^{N_p-1}\ell(x(j)-x_s,\kappa(x(j),y_s)-u_s)\\
&+V_f(x(N_p)-x_s,y_s)+V_O(y_s-y_t)
\end{aligned}
\tag{8}
$$

现对目标函数的终端代价进行修改，修改为加权代价：$\gamma V_f(x-x_s,ys)$。

进行这种修改只是为了推导出最优的无终端约束的价值函数。

接下来定义五终端约束的优化问题：

Then the optimization problem without terminal constraint, $P^{\gamma}_{N_c,N_p}(x,y_t)$, is given by:

$$
\begin{aligned}
&\min_{\mathbf{u},y_s} \quad V^{\gamma}_{N_c,N_p}(x,y_t;\mathbf{u},y_s) && \text{(12a)}\\
&s.t. \quad (9b)-(9g) && \text{(12b)}
\end{aligned}
$$

这种修改，类似于硬约束转化为软约束，即通过增加终端代价（cost to go）的权重来达到终端约束的效果。

定义 $V^{\gamma,0}_{N_c,N_p}(x,y_t)$ 为上面优化问题 $P^{\gamma}_{N_c,N_p}(x,y_t)$ 的最优值，定义集合：

$$\Upsilon_{N_p,\gamma}(y_t)=\{x : V^{\gamma,0}_{N_c,N_p}(x,y_t) - V_O(y_s^* - y_t) \leq N_p d + \gamma\alpha\}$$

where d is a positive constant such that $\ell(x - g_x(y_s), u - g_u(y_s)) \geq d$ for all $(x, y_s) \notin \Gamma_\alpha$ (see Lemma 4 in the appendix).

首先要说明 cost function 表达的是跟踪误差，这个集合量化了能够较好进行跟踪的状态 x 的范围。

$N_p d$ 表示不变集中跟踪误差的上限，$\gamma\alpha$ 表示不变集中终端代价的上限。

在这个集合内部的所有状态 x，作为初始状态时，系统的稳定性都能够得到保证。

γ 越大，集合 $\Upsilon_{N_p,\gamma}(y_t)$ 代表的稳定区域越大，文中没有证明，直观理解就是 γ 越大，终端代价权重越大，“约束”效果越强，会让更多范围的初始状态进入不变集。

接下来就是计算参数 γ 。

Property 2: Assume that $\mathcal{Z}$ is a compact set. Let D be a constant such that $\ell(x - x_s, u - u_s) \leq D$ for all $(x, u) \in \mathcal{Z}$ and $(x_s, u_s) \in \mathcal{Z}_s$. Let $\hat{V}_O$ be a constant such that $V_O(y_s - y_t) \leq \hat{V}_O$ for all $y_s \in \mathcal{Y}_t$ and for all possible y_t. Define the constant γ_0 as

$$\gamma_0 = \max\left(\frac{N_c D - N_p d + \hat{V}_O}{\alpha}, 1\right)$$

Then

(i) $\Upsilon_{N_p,\gamma_0}(y_t)$ contains the domain of attraction of the controller with terminal equality constraint, i.e. $\mathcal{X}_{N_c}$

(ii) For any $\gamma \geq \gamma_0$, the set $\Upsilon_{N_p,\gamma}(y_t)$ contains the domain of attraction of the controller with a prediction and control horizon equal to N_c and the terminal constraint set $\Gamma_{\rho\alpha}$, where $\rho = 1 - \frac{\gamma_0}{\gamma}$.

避障功能

Rossister 的参考前馈 MPC 效果不是很好：

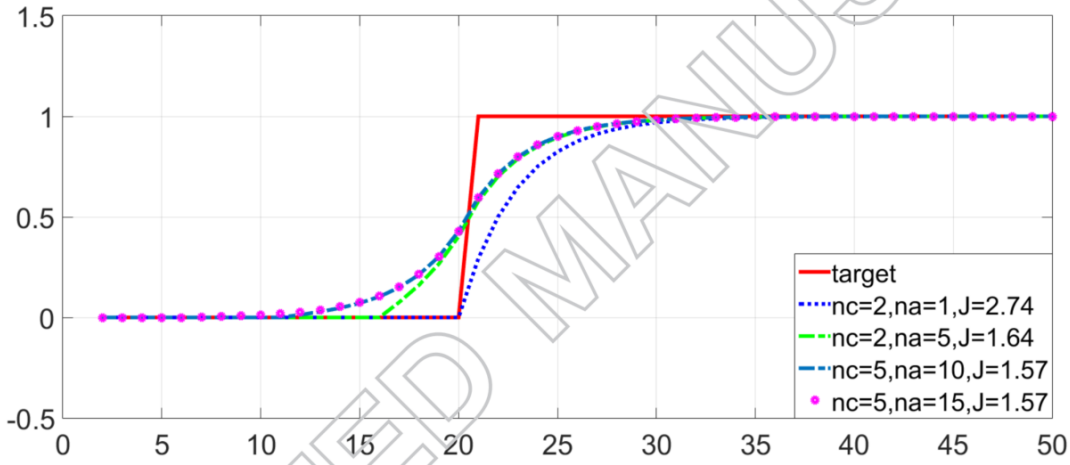

Figure 2. Closed-loop step responses for system (20) with various n_a, n_c.

对于“**避障/防止碰撞**”的要求，使用障碍函数即可。