CRICOS
00213J

ENN582: Reinforcement Learning and Optimal Control

Model Predictive Control

Daniel E. Quevedo[‡]

VERSION 1.

Abstract

The purpose of these notes is to provide a very brief introduction into Model Predictive Control. There exist many textbooks on this topic. For example, [6, 2] provide good introductions, whereas [9, 4] delve further into more advanced aspects.

1 Key Learning Objectives

Key Content Points:

- Model Predictive Control Fundamentals
- Linear Time-Invariant Systems and Quadratic Costs
- Stability Issues
- Application to Vehicle Platooning

[‡]Please report errors within this document to daniel.quevedo@qut.edu.au

Key Learning Objectives - in conjunction with the practical:

- Understand the principles underpinning model predictive control
- Be able to design model predictive controllers and apply them to practical situations
- (optional) Understand how optimality of model predictive control ensures closed-loop stability

2 Model Predictive Control and Receding Horizon Optimisation

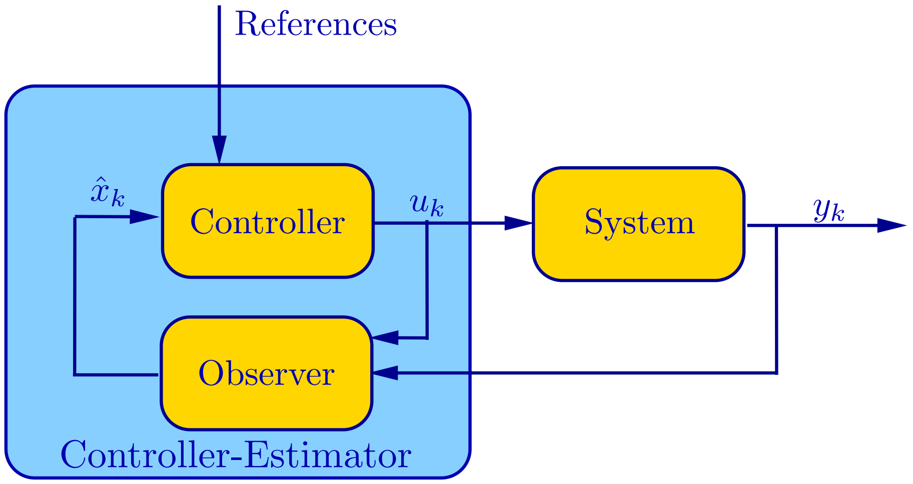

Figure 1: Control loop comprising a controller and a state observer (estimator)

Model Predictive Control (MPC), at times called Receding Horizon Control, is a very powerful state-feedback control method.

- This technique can be used for a wide class of systems, e.g., exhibiting non-linear dynamics, with input and state-constraints. System models can be deterministic or stochastic, or may even be simulation-based
- The main idea consists in calculating the system input through an on-line optimisation.

Basic concepts

To fix ideas, we shall focus on a (deterministic) state-space model,

$$x_{k+1} = f(x_k, u_k), \quad k \in \mathbb{N}_0, \quad f(0,0) = 0$$

with constrained states $x_k \in \mathbb{X} \subseteq \mathbb{R}^n$ and constrained inputs $u_k \in \mathbb{U} \subseteq \mathbb{R}^m$

Consider a cost function of the form

$$J(x_k, \vec{u}) = \sum_{\ell=0}^{N-1} L(x_{k+\ell|k}, u_{k+\ell|k}) + F(x_{k+N|k}) \tag{1}$$

This class of cost function examines predictions of the system model over a finite horizon of length N, as per:

$$x_{k+\ell+1|k} = f(x_{k+\ell|k}, u_{k+\ell|k}) \in \mathbb{X} \tag{2}$$

with $x_{k|k} = x_k$ the (measured) system state, while the entries in

$$\vec{u} \triangleq \begin{bmatrix} u_{k|k} \\ u_{k+1|k} \\ \vdots \\ u_{k+N-1|k} \end{bmatrix} \in \mathbb{U}^N \tag{3}$$

are the associated plant inputs (to be designed/calculated).

In (1), the stage costs $L(\cdot,\cdot)$ can be used to penalise predicted state values and system inputs, whereas $F(\cdot)$ can at times be be used to ensure closed loop stability.[1] These functions, in conjunction with the horizon length N, constitute the main design parameters of a model predictive controller. Often the cost function is chosen to be quadratic, in which case:

$$L(x, u) = x^T Q x + u^T R u, \quad F(x) = x^T Q_F x$$

so that

$$J(x_k, \vec{u}) = \sum_{\ell=0}^{N-1} \left(x_{k+\ell|k}^T Q x_{k+\ell|k} + u_{k+\ell|k}^T R u_{k+\ell|k} \right) + x_{k+N|k}^T Q_F x_{k+N|k} \tag{4}$$

In the above expressions, the matrices Q and Q_F are positive semi-definite (notation $Q \geq 0$, $Q_F \geq 0$) and R is positive definite (notation $R > 0$). With this parameterisation of the cost function, one can choose large values of the matrix Q in comparison to R if one wishes to drive the system state to the origin quickly, at the expense of having to use large control inputs. Conversely, small values of Q relative to R, will generally lead to slow state trajectories, but also only require small control actions.

More general choices of cost functions, are possible. The cost function should reflect what the designer wants to achieve.

Receding horizon optimisation

Minimization at time k and given x_k gives the optimiser:

$$\vec{u}_k^* = \begin{bmatrix} u_{k|k}^* \\ u_{k+1|k}^* \\ \vdots \\ u_{k+N-1|k}^* \end{bmatrix} \triangleq \text{argmin } J(x_k, \vec{u}); \quad \vec{u} \in \mathbb{U}^N,$$

[1]This will be examined further in Section 4.

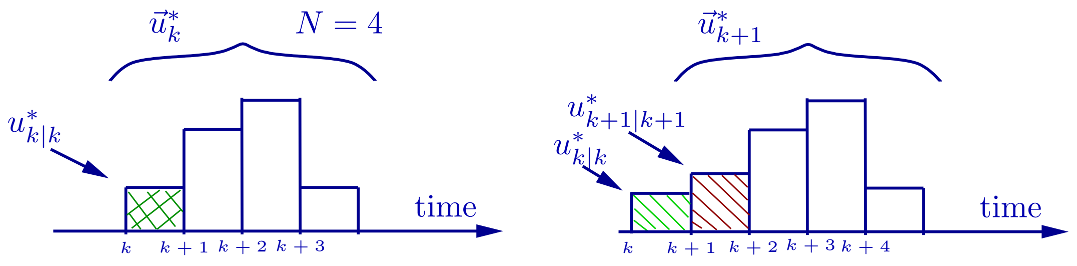

Figure 2: Receding-horizon Optimisation

which depends on system dynamics, cost function, constraints and on the current system state.

Note that $\vec{u}_k^*$ contains values which could be used at the plant input over the entire horizon; from current time k to $k+N-1$. However, only the first element is used; i.e., one sets:

$$u_k \leftarrow \begin{bmatrix} I_m & 0_m & \dots & 0_m \end{bmatrix} \vec{u}_k^*$$

$$u_k \leftarrow u_{k|k}^*$$

At the next time instant, $k+1$, using the system state x_{k+1}, the procedure is repeated, leading to

$$\vec{u}_{k+1}^* = \underset{\vec{u} \in \mathbb{U}^N}{\operatorname{argmin}} J(x_{k+1}, \vec{u})$$

and the plant input

$$u_{k+1} \leftarrow \begin{bmatrix} I_m & 0_m & \dots & 0_m \end{bmatrix} \vec{u}_{k+1}^* = \vec{u}_{k+1|k+1}^*$$

Comments

1. Model predictive controllers can, in principle, be used for 'any' dynamical system model. It has received considerable attention in both academia and industry.

2. Typically longer horizons N give better performance, since the longer-term impact of the current decision u_k is explicitly taken into account. However, often horizons of moderate length give "near" infinite horizon optimal performance. This is illustrated in Figure 3.

3. For nonlinear system models typically numerical optimisation routines need to be used. Especially for large horizons N, this may require significant computations.

4. MPC implicitly defines a state feedback control policy (as in Figure 1):

$$\kappa : \mathbb{X} \to \mathbb{U}$$
$$x_k \to u_k = \kappa(x_k)$$

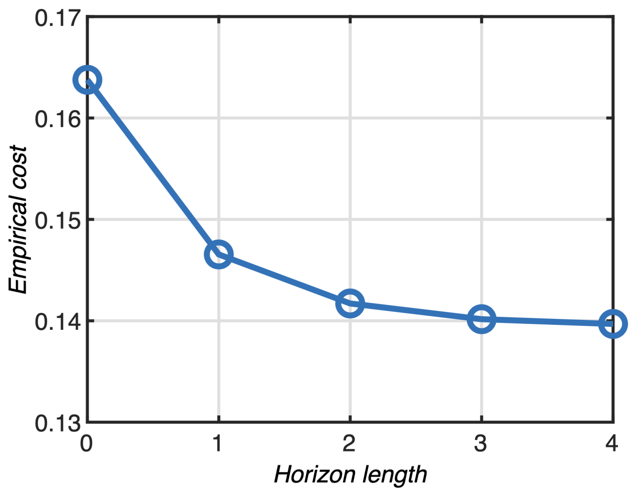

Figure 3: Effect of horizon length on achieved performance, when using model predictive control methods for signal coding. For more information, see [8].

5. Integral action can be included, e.g., by penalizing control increments, $u_{k+\ell+1|k} - u_{k+\ell|k}$, in the cost function (1).

6. To ensure that state constraints (2) are satisfied, decision variables $\vec{u}$ may have to be constrained to belong to a subset of $\mathbb{U}$. When both state and input constraints are present, at times the problem may become infeasible.

3 LTI Systems and Quadratic Cost

In the linear quadratic case, i.e. where the cost function is given by (4), the system dynamics satisfies $f(x,u) = Ax + Bu$ and the inputs and outputs are unconstrained ($\mathbb{X} = \mathbb{R}^n$, $\mathbb{U} = \mathbb{R}^m$), model predictive control is equivalent to implementing the first element of the finite-horizon LQR optimizer.

Controller tuning

A common choice for the weighting matrices is $Q = Q_F = C^T C$, $R = \rho I$, where $\rho > 0$ and where the system "output" is given by $y_k = Cx_k$. Then, the cost can be rewritten as per:

$$J(x_k, \vec{u}) = \sum_{\ell=0}^{N} \| \, y_{k+\ell|k} \, \|^2 + \rho \sum_{\ell=0}^{N-1} \| \, u_{k+\ell|k} \, \|^2$$

and ρ gives relative weighting of the output energy, say J_{out}, and the input energy, say J_{out}. These are competing objectives. Ideally, both quantities should be small.

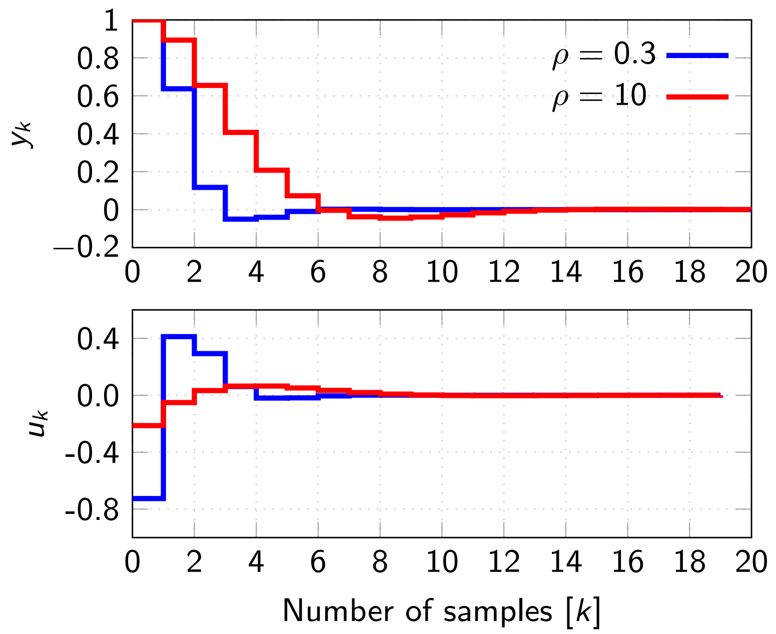

Figure 4: Input and output trajectories

As an example, consider a vehicle of mass $m = 1$ moving in a one-dimensional space under the effect of a time-varying force input $u(t)$. This can be modelled as a double integrator system:

$$\begin{aligned}\dot{x}(t) &= \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix} x(t) + \begin{bmatrix}0\\ 1/m\end{bmatrix} u(t)\\ y(t) &= \begin{bmatrix}1 & 0\end{bmatrix} x(t).\end{aligned} \tag{5}$$

In the above, the first element of the state corresponds to the vehicle position, whereas the second element of $x(t)$ amounts to its velocity. Position measurements are available.

A discrete-time model can be obtained, e.g., using a series expansion. This leads to

$$\begin{aligned}x_{k+1} &= \begin{bmatrix}1 & \Delta t\\ 0 & 1\end{bmatrix} x_k + \begin{bmatrix}\frac{1}{2}(\Delta t)^2\\ \Delta t,\end{bmatrix} u_k\\ y_k &= \begin{bmatrix}1 & 0\end{bmatrix} x_k,\end{aligned}$$

where Δt is the sampling interval, so that $x_k = x(k\Delta t)$, etc. Figure 4 illustrates input and output trajectories when using a horizon $N = 20$ for two different values of ρ.[2] Larger ρ allow the designer to more significantly penalise control actions, leading to smaller control actions and often slower convergence. More generally, the design parameter ρ allows one to trade off input and output energies, see Figure 5.

[2]The initial state is taken as $x_0 = \begin{bmatrix}1 & 0\end{bmatrix}^T$, The sampling interval is chosen as $\Delta t = 1$.

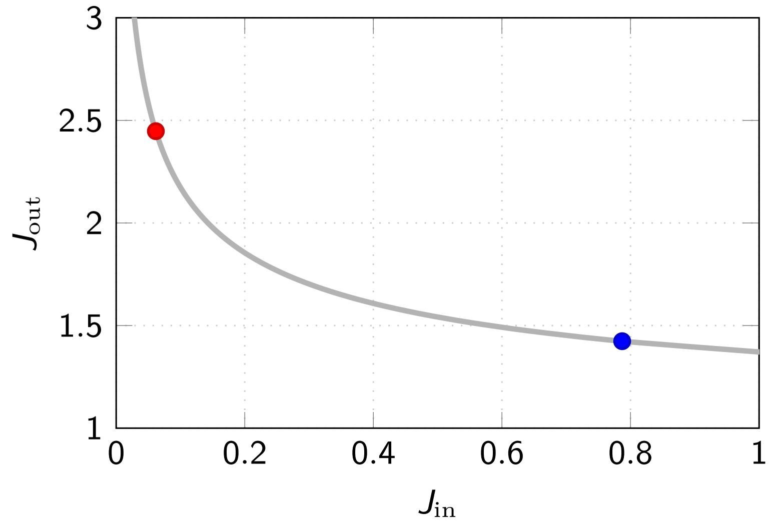

Figure 5: Empirical trade-off curve between input and output energies. The blue circle corresponds to $\rho = 0.3$, whereas the red circle denotes a design with $\rho = 10$.

Explicit solution

Since $x_{k+1} = Ax_k + Bu_k$, state predictions are explicitly given by:

$$x_{k+\ell|k} = A^\ell x_k + \sum_{j=0}^{\ell-1} A^{\ell-j-1} B u_{k+j|k}$$

It is convenient to define

$$\vec{x}_k = \begin{bmatrix} x_{k+1|k} \\ \vdots \\ x_{k+N|k} \end{bmatrix}$$

so that

$$\vec{x}_k = \Phi \vec{u} + \Lambda x_k,$$

where:

$$\Phi = \begin{bmatrix} B & 0 & \dots & \dots & 0 \\ AB & B & & & \\ A^2B & AB & \dots & \dots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & 0 \\ A^{N-1}B & & \dots & AB & B \end{bmatrix} \quad \text{and} \quad \Lambda = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}.$$

The cost function in (4) (with $R = \rho I$) can be rewritten as per;

$$J(x_k, \vec{u}) = x_k^T Q x_k + \vec{x}_k^T \bar{Q} \vec{x}_k + \rho \vec{u}^T \vec{u} = x_k^T Q x_k + (\Phi \vec{u} + \Lambda x_k)^T \bar{Q} (\Phi \vec{u} + \Lambda x_k) + \rho \vec{u}^T \vec{u}$$

$$= x_k^T (\Lambda^T \bar{Q} \Lambda + Q) x_k + \vec{u}^T (\Phi^T \bar{Q} \Phi + \rho I_{mN}) \vec{u} + 2 \vec{u}^T \Phi^T \bar{Q} \Lambda x_k,$$

where $\bar{Q} \triangleq \text{diag}(Q, \ldots, Q, Q_F)$. To minimize $J(x_k, \vec{u})$, we compute:

$$\frac{\partial J}{\partial \vec{u}} = 2 \Phi^T \bar{Q} \Lambda x_k + 2 (\Phi^T \bar{Q} \Phi + \rho I_{mN}) \vec{u} = 0 \implies \vec{u}_k^* = -(\Phi^T \bar{Q} \Phi + \rho I_{mN})^{-1} \Phi^T \bar{Q} \Lambda x_k$$

Since $\rho > 0$, the Hessian is positive definite and the optimizer is given by the above.

Therefore we obtain a constant gain state feedback controller.

$$\begin{aligned} u_k &= -K x_k \\ K &= \begin{bmatrix} I_m & 0_m & \ldots & 0_m \end{bmatrix} (\Phi^T \bar{Q} \Phi + \rho I_{mN})^{-1} \Phi^T \bar{Q} \Lambda \end{aligned}$$

Not surprisingly, the above control law is equivalent to that provided by a finite-horizon Linear Quadratic Regulator (LQR). We note that LQR with an infinite horizon is often preferable, since the resulting control law is time-invariant. An example will be given in Section 5.

For systems with linear dynamics, but where inputs or states are constrained to belong to convex sets, MPC optimisation problems with a quadratic cost can be cast as quadratic programmes. The latter can be tackled using efficient optimisation routines, for details see, e.g., [4, 6]. Such a scenario is explored in Section 5.

4 Stability Analysis (optional)

We will return to the general formulation of Section 2 and show how the cost function can be designed to ensure attractivity of the origin, i.e.[3]

$$\forall x_0 \in \mathbb{R}^n, \qquad \lim_{k \to \infty} \|x_k\| = 0$$

Assumptions

1. The cost function terms satisfy:

$$\begin{aligned} F(0) = 0, &\quad F(x) \geq 0, \quad \forall x \in \mathbb{X} = \mathbb{R}^n, \\ L(0,0) = 0, &\quad L(x,u) \geq \alpha(\|x\|), \end{aligned}$$

   where $\alpha(\cdot)$ is a monotonically increasing function and satisfies $\alpha(0) = 0, \lim_{s \to \infty} \alpha(s) = \infty$.

[3]For a more careful and thorough treatment, see [9].

2. There exists a control law $\kappa_f : \mathbb{R}^n \to \mathbb{U}$ which stabilises the system f, in a sense that
$$F(f\big(x, \kappa_f(x)\big) \leq F(x) - L(x, \kappa_f(x))$$

The above two assumptions ensure that the origin is attractive for the artificial system[4]

$$x_{k+1} = f\big(x_k, \kappa_f(x_k)\big) \tag{6}$$

This raises the question on whether the use of MPC, $x_{k+1} = f\big(x_k, \kappa(x_k)\big)$, also ensures attractivity of the origin?

In fact, it turns out that if these assumptions are satisfied, then the origin ($x = 0$) is also attractive, when MPC is used! This can be established as follows: Let us consider the sequence of optimal costs:

$$V_k \triangleq J(x_k, \vec{u}_k^*), \qquad k \in \mathbb{N}_0$$

where, as before

$$\vec{u}_k^* = \begin{bmatrix} u_{k|k}^* \\ u_{k+1|k}^* \\ \vdots \\ u_{k+N-1|k}^* \end{bmatrix} \in \mathbb{U}^N, \qquad \mathbb{U} \subset \mathbb{R}^m$$

We also denote the associated (optimal at time k) state trajectory via

$$\vec{x}_k^* = \begin{bmatrix} x_{k+1|k}^* \\ x_{k+2|k}^* \\ \vdots \\ x_{k+N|k}^* \end{bmatrix}$$

Note that:

$$\vec{u}_{k+1}^* = \begin{bmatrix} u_{k+1|k+1}^* \\ u_{k+2|k+1}^* \\ \vdots \\ u_{k+N|k+1}^* \end{bmatrix},$$

and typically, we have $u_{k+1|k}^* \neq u_{k+1|k+1}^*$.

We shall show that $\{V_k\}$, $k \in \mathbb{N}_0$ is non-increasing. For that purpose, note that, due to optimality, we have

$$V_{k+1} \leq J(x_{k+1}, \vec{u}^0), \quad \forall \vec{u}^0 \in \mathbb{U}^N.$$

[4]This can be shown by considering $F(\cdot)$ as a Lyapunov function for (6).

In particular, let us choose

$$\vec{u}^0 = \begin{bmatrix} u^*_{k+1|k} \\ u^*_{k+2|k} \\ \vdots \\ u^*_{k+N-1|k} \\ u^0 \end{bmatrix} \quad \text{where } u^0 = \kappa_f(x^*_{k+N|k})$$

This suboptimal sequence is formed by reusing outcomes of the optimisation at time k, which led to V_k. This choice allows us to compare V_k with $J(x_{k+1}, \vec{u}^0)$. Note that, by definition,

$$V_k = J(x_k, \vec{u}^*_k) = L(x_k, u^*_{k|k}) + L(x^*_{k+1|k}, u^*_{k+1|k}) + \dots \quad + L(x^*_{k+N-1|k}, u^*_{k+N-1|k}) + F(x^*_{k+N|k})$$

Implementation of the first element of the optimiser $\vec{u}^*_k$ leads to the state value

$$x_{k+1} = f(x_k, u^*_{k|k})$$

Thus, the state sequence associated to $\vec{u}^0$ is given by

$$\{f(x_k, u^*_{k|k}),\ x^*_{k+2|k}, \dots,\ x^*_{k+N|k},\ f\big(x^*_{k+N|k}, \kappa_f(x^*_{k+N|k})\big)\}$$

Direct calculations show that

$$\begin{aligned} &J(x_{k+1}, \vec{u}^0) = J(x^*_{k+1|k}, \vec{u}^0) \\ &= L(x^*_{k+1|k}, u^*_{k+1|k}) + L(x^*_{k+2|k}, u^*_{k+2|k}) + \dots + L(x^*_{k+N|k}, u^0) + F\big(f(x^*_{k+N|k}, u^0)\big) \\ &\quad + \Big(L(x_k, u^*_{k|k}) + F(x^*_{k+N|k}) - L(x_k, u^*_{k|k}) - F(x^*_{k+N|k})\Big) \\ &= V_k - L(x_k, u^*_{k|k}) - F(x^*_{k+N|k}) + L\big(x^*_{k+N|k}, \kappa_f(x^*_{k+N|k})\big) + F\big(f(x^*_{k+N|k}, \kappa_f(x^*_{k+N|k}))\big) \end{aligned}$$

where $u^0 = \kappa_f(x^*_{k+N|k})$

Using Assumption 2, we have $-F(x) + L(x, \kappa_f(x)) + F(f(x, \kappa_f(x)) \leq 0$, and thus

$$J(x_{k+1}, \vec{u}^0) \leq V_k - L(x_k, u^*_{k|k})$$

Since $V_{k+1} \leqslant J_{k+1}(x_{k+1}, \vec{u}^0)$, we conclude that $V_{k+1} \leqslant V_k - L(x_k, u^*_{k|k})$

Assumption 1 gives $F(x) \geq 0$, $L(x,u) \geq 0$, and $-L(x,u) \leq -\alpha(||x||)$, which implies

$$0 \leqslant V_{k+1} \leqslant V_k - \alpha(||x_k||)$$

As a consequence, the sequence $\{V_k\}, k \in \mathbb{N}$ is non-increasing and bounded from below. Therefore $\{V_k\}$ converges as $k \to \infty$.

Consequently,

$$\lim_{k\to\infty} \alpha(||x_k||) = 0 \Rightarrow \lim_{k\to\infty} ||x_k|| = 0.$$

This establishes the desired stability result.

Linear quadratic control with finite set inputs

As a special instance, consider

$$\begin{cases} L(x,u) = x^T Q x + u^T R u, & Q > 0, R \geqslant 0 \\ F(x) = x^T Q_F x \end{cases}$$

The plant mode is given by $f(x,u) = Ax + Bu$, where u is restricted to belong to the finite set

$$\mathbb{U} = \{S_1, S_2 \ldots, S_G\}; \quad S_1 \ldots S_G \in \mathbb{R}^m, \quad 0 \in \mathbb{U}.$$

This class of controller can be used, e.g., for power conversion [7]; Heating, ventilation, and air conditioning systems; source coding (A/D conversion), etc.

If we choose Q_F to satisfy the Lyapunov equation[5]

$$A^T Q_F A + Q = Q_F,$$

then MPC stabilises this non-linear system. In fact, Assumption 1 is clearly satisfied, since

$$\begin{array}{ll} F(x) = x^T Q_F x \geqslant 0 & L(0,0) = 0 \\ F(0) = 0 & L(x,u) \geqslant x^T Q x \geqslant \underbrace{\lambda_{\min}(Q)}_{>0} x^T x = \lambda_{\min}(Q)||x||^2 \end{array}$$

Thus, we can take $\alpha(s) = \lambda_{\min}(Q)s^2$

To verify Assumption 2, choose $\kappa_f(x) = 0 \; \forall x \in \mathbb{R}^n$

$$\begin{aligned} F\big(f(x,\kappa_f(x))\big) &= F(Ax) = (Ax)^T Q_F (Ax) = x^T A^T Q_F A x = x^T (Q_F - Q)x \\ &= F(x) - L(x, \kappa_f(x)) \end{aligned}$$

We conclude that if MPC is used, then $x_k \to 0$

5 Case Study: Vehicle Platooning (optional)

Autonomous vehicle platooning permits road vehicles to travel very close together increasing road capacity while reducing vehicle energy use. Standard Cruise Control design operate only using on board information to maintain a vehicle at a set velocity. Adaptive Cruise Control uses on board sensors to measure the distance and velocity of a predecessor vehicle from a follower vehicle to maintain either a desired velocity or safe distance between the predecessor and follower vehicles. Adaptive Cruise Control designs, while often stable in a closed-loop sense, may exhibit the effect known as string instability.

[5]Note that this requires that A be a stable matrix

String instability occurs when a disturbance on a vehicle is amplified through other vehicles in the platoon [3]. Such a disturbance could be the result of a pedestrian stepping out onto the road causing a driver to brake. To achieve string stability, a more complicated control design is required. This could involve a nonlinear inter-vehicle spacing policies, utilising information from other vehicles in the platoon, or incorporating future control action. Cooperative Adaptive Cruise Control designs utilise inter-vehicle communication to share measurements, state information, and intended control actions with other vehicles in the platoon.

In the below, we will investigate a follower control design where state information and control actions are reliably shared with the follower vehicle. The approach below designs the control action at each follower vehicle only taking into account the control action of the predecessor vehicle. An alternative approach is to jointly design the control action of every vehicle using the aggregated "platoon state," that combines the states of all vehicles, see [5].

Single dimension vehicle dynamics

The one-dimensional forward acceleration of a vehicle-i as a function of the control action is often modelled with a first order filter

$$a_i(s) = \frac{1}{\tau_i s + 1} u_i(s)$$

where $a_i(t)$ is the acceleration and τ_i is the mechanical actuation lag. The mechanical actuation lag τ_i describes the time delay between a change in the desired control action and the change in response in the vehicle. This is a simple modelling technique to represent the time delay resulting from both the engine and the braking systems. Often we consider a homogeneous platoon where all vehicles have the same actuation lag $\tau = \tau_i$, particularly when the platoon design is intended to have a similar style of vehicle. For a standard passenger vehicle the lag is approximately $\tau \approx 1.5$.

Taking the inverse Laplace transform, we can find that the time domain vehicle state can be described by the following integrator model

$$\begin{aligned}
\dot{p}_i(t) &= v_i(t) \\
\dot{v}_i(t) &= a_i(t) \\
\dot{a}_i(t) &= -\frac{1}{\tau} a_i(t) + \frac{1}{\tau} u_i(t)
\end{aligned}$$

where $p_i(t)$ is the position and $v_i(t)$ is the velocity.

Let the continuous time state be $x_i(t) = [p_i(t), v_i(t), a_i(t)]^\mathsf{T}$, then the dynamics can be written as:

$$\dot{x}_i(t) = A_c x_i(t) + B_c u_i(t)$$

where

$$A_c = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1/\tau_i \end{bmatrix} \quad \text{and} \quad B_c = \begin{bmatrix} 0 \\ 0 \\ 1/\tau_i \end{bmatrix}$$

Many vehicle platoons designs use this model, or a reduced integrator model of only position and velocity.

This model be converted to discrete-time as many controllers are typically implemented in discrete-time, such as MPC. The state and control actions are indexed by the discrete-time step k where the real time $t = kT_s$ with sampling period T_s. We write the state in discrete time as $x_{i,k} = [p_{i,k}, v_{i,k}, a_{i,k}]^\mathsf{T}$ to give the discrete-time dynamics

$$x_{i,k+1} = Ax_{i,k} + Bu_{i,k} + w_{i,k}$$

We can convert to discrete-time by $A = \exp(A_c T_s)$ and $B = \int_0^{T_s} \exp(A_c m) dm B_c$, which can be efficiently computed by [1]:

$$\begin{bmatrix} A & B \\ 0 & I \end{bmatrix} = \exp\left(\begin{bmatrix} A_c & B_c \\ 0 & 0 \end{bmatrix} T_s \right)$$

Follower control design

We seek to control the inter-vehicle distance between two vehicles by controller the follower vehicle using an LQR control design. Let us assume that the state and control action of the predecessor vehicle is exactly known to the follower vehicle.

Let us write the error or difference between two vehicles (vehicle-$i-1$) and the follower vehicle (vehicle-i) over time

$$e_{i,k} = x_{i-1,k} - x_{i,k}.$$

We will use this error signal to design our follower controller.

Using the discrete-time state-space dynamics, let us derive the dynamics of the error signal.

Start with the error at time $k+1$ and use the vehicle dynamics

$$\begin{aligned} e_{i,k+1} &= x_{i-1,k+1} - x_{i,k+1} \\ &= Ax_{i-1,k} + Bu_{i-1,k} + w_{i-1,k} - Ax_{i,k} - Bu_{i,k} - w_{i,k} \\ &= A(x_{i-1,k} - x_{i,k}) + B(u_{i-1,k} - u_{i,k}) + w_k \\ &= Ae_{i,k} + B\bar{u}_k + w_k \end{aligned}$$

where $w_k = w_{i-1,k} - w_{i,k}$ and $\bar{u}_{i,k} = u_{i-1,k} - u_{i,k}$. Fortunately, by assumption that the vehicles have the same dynamics, the error signal also has the same dynamics.

The control action for the follower vehicle-i is the combination of the known control action $u_{i-1,k}$ of the predecessor vehicle-$i-1$ and the control action $\bar{u}_{i,k}$ to reduce the error signal signal $e_{i,k}$:

$$u_{i,k} = u_{i-1,k} - \bar{u}_{i,k}.$$

The control action of the lead vehicle $u_{1,k}$ is an exogenous input to the system.

We will explore how to design the error signal control action $\bar{u}_{i,k}$.

LQR cost function design

We start with infinite horizon LQR design. Let us write a cost function of the state at the current time[6] k to penalise the quadratic error between the two vehicles to a given distance and the quadratic control action:

$$J(e_{i,k}, \bar{u}_{i,k}) = \sum_{\ell=0}^{\infty} q_1(p_{i-1,k+\ell} - p_{i,k+\ell} - p^\star)^2 + q_2(v_{i-1,k+\ell} - v_{i,k+\ell} - v^\star)^2 \\ + q_3(a_{i-1,k+\ell} - a_{i,k+\ell} - a^\star)^2 + R\bar{u}_{i,k+\ell}^2$$

where $p^\star$, $v^\star$, and $a^\star$ are the desired inter-vehicle distance, velocity, and acceleration , and q_1, q_2, and q_3 are the penalty on the position difference, velocity difference, and acceleration difference, and R is the penalty on control actions.

Our cost function can be written in the convenient form

$$J(e_{i,k}, \bar{u}_{i,k}) = \sum_{\ell=0}^{\infty} (e_{i,k+\ell} - e^\star)^\mathsf{T} Q (e_{i,k+\ell} - e^\star) + \bar{u}_{i,k+\ell}^\mathsf{T} R \bar{u}_{i,k+\ell}$$

where $e^\star$ is a reference vector $e^\star = [p^\star, v^\star, a^\star]^\mathsf{T}$ and Q is a diagonal matrix $Q = \text{diag}\{q_1, q_2, q_3\}$.

The optimal control action is

$$\bar{u}_{i,k}^\star = -K(e_{i,k} - e^\star)$$

where the optimal gain is K and S is the solution to the discrete-time algebraic Riccati equation:

$$\begin{aligned} K &= (R + BSB^\mathsf{T})^{-1} B^\mathsf{T} PA \\ S &= A^\mathsf{T} SA - A^\mathsf{T} SB(R + BSB^\mathsf{T})^{-1} B^\mathsf{T} SA + Q. \end{aligned}$$

The control action on vehicle-i is then

$$u_{i,k} = u_{i-1,k} + K(e_{i,k} - e^\star).$$

Numerical illustration

Let us quickly consider a numerical illustration of three vehicles. We consider a mechanical actuation lag of $\tau = 1.5$. We define our reference as $e^\star = [1, 0, 0]^\mathsf{T}$, and cost as $Q = \text{diag}\{100, 1, 1\}$ and $R = 10^{-3}$.

Figure 6 shows the difference in position between the vehicles (left: with the difference between vehicle-1 and-2 in blue, and difference between vehicle-2 and-3 in yellow) and velocities (right: vehicle 1 in blue, 2 in yellow, 3 in green). Inspecting the simulation output we observe that difference between vehicles-2 and-3 becomes negative. Vehicle-3 crashes into vehicle-2! Additionally, vehicle-2 will reach a velocity of over 40 m/s (144 km/h), while vehicle-3 reaches over 80 m/s (288 km/h), which are absurd speeds.

[6]For simplicity we have used a simplified subscript notation. The notation should be the prediction at time $k + \ell$ given the state at time k or $k + \ell|k$.

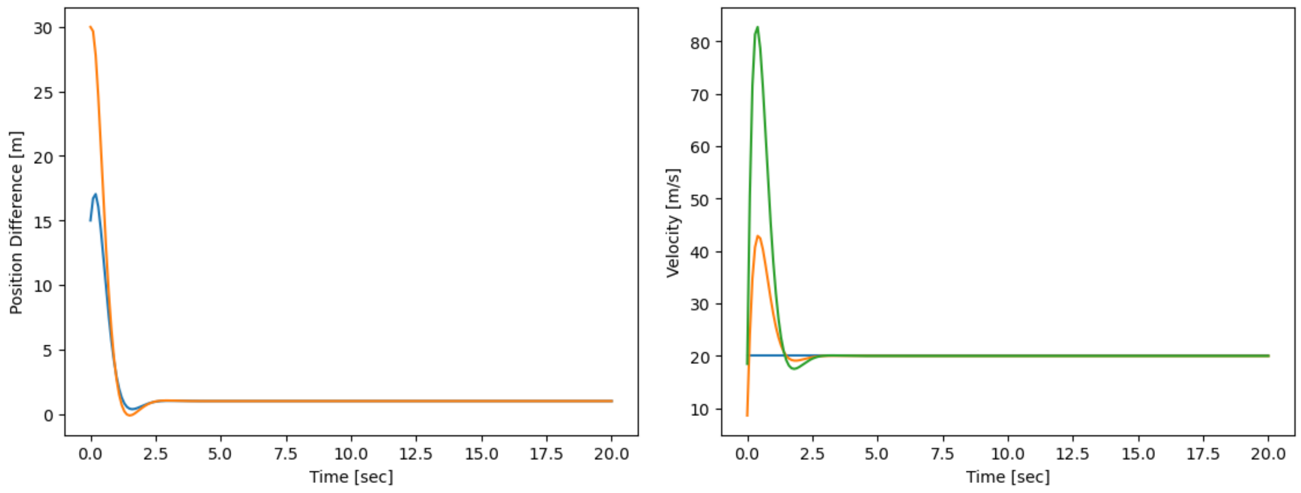

Figure 6: Numerical illustration of a platoon of three vehicles.

From this numerical illustration, as a control designer we would be concerned about inter-vehicle distance safety, legal velocity limits, and vehicle actuation limits.

Model Predictive Control

In MPC we utilise our mathematical model to predict the state into the future to decide our control actions. By inspecting these predictions of the state, we can identify if our choice of control action sequence through the dynamics will result in the state moving into an unsafe or undesired region. We can directly include constraints on both the state and control action as part of the control action optimisation process, so that control decisions now can prevent a future issue.

To illustrate, we now refine our control design from above. We start by taking our infinite horizon cost function from our LQR implementation and write as a finite horizon cost function. We introduce reasonable state constraints, before computing our optimal control action that both minimises the cost function and satisfies the constraints.

We recall that vehicle-i have the dynamics

$$x_{i,k+1} = Ax_{i,k} + Bu_{i,k}$$

and the inter-vehicle error model has the form

$$e_{i,k+1} = Ae_{i,k} + Bu_{i-1,k} - Bu_{i,k}$$

then state predictions are explicitly given by:

$$\begin{aligned}
x_{i,k+\ell|k} &= A^{\ell} x_{i,k} + \sum_{j=0}^{\ell-1} A^{\ell-j-1} B u_{i,k+j|k} \\
e_{i,k+\ell|k} &= A^{\ell} e_{i,k} + \sum_{j=0}^{\ell-1} A^{\ell-j-1} B u_{i-1,k+j|k} - B u_{i,k+j|k}.
\end{aligned}$$

It is convenient to define

$$\vec{e}_{i,k} = \begin{bmatrix} e_{i,k+1|k} \\ \vdots \\ e_{i,k+N|k} \end{bmatrix}, \quad \vec{x}_{i,k} = \begin{bmatrix} x_{i,k+1|k} \\ \vdots \\ x_{i,k+N|k} \end{bmatrix}, \quad \text{and} \quad \vec{u}_{i,k} = \begin{bmatrix} u_{i,k+1|k} \\ \vdots \\ u_{i,k+N|k} \end{bmatrix}$$

so that

$$\vec{x}_{i,k} = \Lambda x_{i,k} + \Phi \vec{u}_{i,k}, \tag{7}$$

$$\vec{e}_{i,k} = \Lambda e_{i,k} + \Phi \vec{u}_{i-1,k} - \Phi \vec{u}_{i,k}, \tag{8}$$

where:

$$\Phi = \begin{bmatrix} B & 0 & \dots & \dots & 0 \\ AB & B & & & \\ A^2B & AB & \dots & \dots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & 0 \\ A^{N-1}B & & \dots & AB & B \end{bmatrix} \quad \text{and} \quad \Lambda = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix}.$$

Cost function design

Let us write a finite horizon cost function of the state at the current time[7] k to penalise the quadratic error between the two vehicles to a given distance and the quadratic control action:

$$J(e_{i,k}, \vec{u}) = \sum_{\ell=0}^{N-1} \Big((e_{i,k+\ell} - e^{\star})^{\mathsf{T}} Q (e_{i,k+\ell} - e^{\star}) + \bar{u}_{i,k+\ell}^{\mathsf{T}} R \bar{u}_{i,k+\ell} \Big) + (e_{i,k+N} - e^{\star})^{\mathsf{T}} Q (e_{i,k+N} - e^{\star})$$

where $e^{\star}$, Q, and R are as above. The design of the cost function is inspired by our infinite horizon cost function.

[7]Again, for simplicity we have used a simplified subscript notation. The notation should be the prediction at time $k+\ell$ given the state at time k or $k+\ell|k$.

Let us define $\vec{e}^\star \triangleq [(e^\star)^\intercal \dots (e^\star)^\intercal]^\intercal$. Our cost function can then be rewritten as

$$\begin{aligned}
J(e_{i,k}, \vec{u}) &= (\vec{e}_{i,k} - \vec{e}^\star)^\intercal \bar{Q}(\vec{e}_{i,k} - \vec{e}^\star) + \vec{u}_{i,k}^\intercal \bar{R}\vec{u}_{i,k} + (e_{i,k} - e^\star)^\intercal Q(e_{i,k} - e^\star) \\
&= (\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k} - \Phi\vec{u}_{i,k} - \vec{e}^\star)^\intercal \bar{Q}(\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k} - \Phi\vec{u}_{i,k} - \vec{e}^\star) \\
&\quad + \vec{u}_{i,k}^\intercal \bar{R}\vec{u}_{i,k} + (e_{i,k} - e^\star)^\intercal Q(e_{i,k} - e^\star) \\
&= \vec{u}_{i,k}^\intercal(\Phi^\intercal \bar{Q}\Phi + \bar{R})\vec{u} + 2\vec{u}^\intercal(-\Phi)^\intercal \bar{Q}(\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k} - \vec{e}^\star) \\
&\quad + (\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k} - \vec{e}^\star)^\intercal \bar{Q}(\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k} - \vec{e}^\star) + (e_{i,k} - e^\star)^\intercal Q(e_{i,k} - e^\star),
\end{aligned}$$

where $\bar{Q} \triangleq \operatorname{diag}(Q, \dots, Q)$ and $\bar{R} \triangleq \operatorname{diag}(R, \dots, R)$.

To obtain our optimal control action $\vec{u}$, we seek to minimise J with respect to the state and control constraints.

Constraints

To avoid vehicles in our platoon crashing into each other, we select a minimum inter-vehicle difference of $p_{\min} = 3m$. To obey road rules, we select a minimum velocity of $v_{\min} = 0m/s$ (no reversing!) and maximum velocity $v_{\max} = 25m/s$. For passenger comfort we permit an absolute acceleration of $8m/s^2$. At most the engine and gearbox can output a maximum forward acceleration of $3m/s^2$ and the braking systems saturates at $-6m/s^2$. We want to apply these state and control constraints at every time step.

We want to choose control actions at time k that ensure that the future state $x_{i,k+\ell|k}$ or inter-vehicle state $e_{i,k+\ell|k}$ for all $\ell = 0, \dots, N$ in the finite horizon remains within the constraints $p_{\min} \leq p_{i-1,k+\ell|k} - p_{i,k+\ell|k} \leq p_{\max}$, $v_{\min} \leq v_{i,k+\ell|k} \leq v_{\max}$, $a_{\min} \leq a_{i,k+\ell|k} \leq a_{\max}$ and the control actions $u_{\min} \leq u_{i,k+\ell|k} \leq u_{\max}$.

Consider the minimum distance constraint applying at each time in the finite prediction horizon

$$\begin{aligned}
\begin{bmatrix} p_{i-1,k+1|k} - p_{i,k+1|k} \\ \vdots \\ p_{i-1,k+N|k} - p_{i,k+N|k} \end{bmatrix} &\geq \begin{bmatrix} p_{\min} \\ \vdots \\ p_{\min} \end{bmatrix} \\
\vec{p}_{i,k} &\geq p_{\min} 1_N \\
1_N \otimes \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \vec{e}_{i,k} &\geq p_{\min} 1_N.
\end{aligned}$$

where 1_N is a column vector of size $N \times 1$, and $\vec{p}_{i,k}$ is a column vector of the position differences, and

$\otimes$ is the Kronecker product[8]. We next apply the prediction model (8)

$$p_{\min}1_N \leq 1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} \vec{e}_{i,k}$$

$$p_{\min}1_N \leq 1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} (\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k} - \Phi\vec{u}_{i,k})$$

$$1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} \Phi\vec{u}_{i,k} \leq 1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} (\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k}) - p_{\min}1_N.$$

Now consider the maximum distance constraint applying at each time in the finite prediction horizon

$$\begin{bmatrix} p_{i-1,k+1|k} - p_{i,k+1|k} \\ \vdots \\ p_{i-1,k+N|k} - p_{i,k+N|k} \end{bmatrix} \leq \begin{bmatrix} p_{\max} \\ \vdots \\ p_{\max} \end{bmatrix}$$

$$\vec{p}_{i,k} \leq p_{\max}1_N$$

$$1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} \vec{e}_{i,k} \leq p_{\max}1_N$$

$$1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} \vec{e}_{i,k} \leq p_{\max}1_N$$

$$1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} (\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k} - \Phi\vec{u}_{i,k}) \leq p_{\max}1_N$$

$$-1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} \Phi\vec{u}_{i,k} \leq -1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} (\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k}) + p_{\max}1_N.$$

The minimum velocity constraint applying at each time in the finite prediction horizon can be incor-

[8]The Kronecker product of two matrices produces $A \otimes B = \begin{bmatrix} a_{1,1}B & \dots & a_{1,M} \\ \vdots & \ddots & \\ a_{N,1}B & \dots & a_{N,M}B \end{bmatrix}$ where $a_{i,j}$ is the element of matrix A in row i column j.

porated as follows:

$$\begin{aligned}
\begin{bmatrix} v_{i-1,k+1|k} - v_{i,k+1|k} \\ \vdots \\ v_{i-1,k+N|k} - v_{i,k+N|k} \end{bmatrix} &\geq \begin{bmatrix} v_{\min} \\ \vdots \\ v_{\min} \end{bmatrix} \\
\vec{v}_{i,k} &\geq v_{\min} 1_N \\
v_{\min} 1_N &\leq 1_N \otimes \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \vec{x}_{i,k} \\
v_{\min} 1_N &\leq 1_N \otimes \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} (\Lambda x_{i,k} + \Phi \vec{u}_{i,k}) \\
-1_N \otimes \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \Phi \vec{u}_{i,k} &\leq 1_N \otimes \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \Lambda x_{i,k} - v_{\min} 1_N
\end{aligned}$$

Now consider the maximum velocity constraint applied at each time in the finite prediction horizon

$$\begin{aligned}
\begin{bmatrix} v_{i-1,k+1|k} - v_{i,k+1|k} \\ \vdots \\ v_{i-1,k+N|k} - v_{i,k+N|k} \end{bmatrix} &\leq \begin{bmatrix} v_{\max} \\ \vdots \\ v_{\max} \end{bmatrix} \\
\vec{v}_{i,k} &\leq v_{\max} 1_N \\
1_N \otimes \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \vec{x}_{i,k} &\leq v_{\max} 1_N \\
1_N \otimes \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} (\Lambda x_{i,k} + \Phi \vec{u}_{i,k}) &\leq v_{\max} \\
1_N \otimes \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \Phi \vec{u}_{i,k} &\leq -1_N \otimes \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \Lambda x_{i,k} + v_{\max} 1_N
\end{aligned}$$

The same can be done for the acceleration, readily to:

$$\begin{aligned}
-1_N \otimes \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Phi \vec{u}_{i,k} &\leq 1_N \otimes \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Lambda x_{i,k} - a_{\min} 1_N \\
1_N \otimes \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Phi \vec{u}_{i,k} &\leq -1_N \otimes \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \Lambda x_{i,k} + a_{\max} 1_N.
\end{aligned}$$

The control action constraints are given by:

$$
\begin{aligned}
-I_N \vec{u}_{i,k} &\leq -u_{\min} 1_N \\
I_N \vec{u}_{i,k} &\leq u_{\max} 1_N,
\end{aligned}
$$

where I_N is the identity matrix of size N.

All of the constraints can be written together in the form $A_{in}\vec{u}_{i,k} \leq B_{in}$

$$
\begin{bmatrix}
1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} \Phi \\
-1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} \Phi \\
-1_N \otimes \begin{bmatrix}0\\1\\0\end{bmatrix} \Phi \\
1_N \otimes \begin{bmatrix}0\\1\\0\end{bmatrix} \Phi \\
-1_N \otimes \begin{bmatrix}0\\0\\1\end{bmatrix} \Phi \\
1_N \otimes \begin{bmatrix}0\\0\\1\end{bmatrix} \Phi \\
-I_N \\
I_N
\end{bmatrix}
\vec{u}_{i,k} \leq
\begin{bmatrix}
1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} (\Lambda e_{i,k} + \Phi \vec{u}_{i-1,k}) - p_{\min} 1_N \\
-1_N \otimes \begin{bmatrix}1\\0\\0\end{bmatrix} (\Lambda e_{i,k} + \Phi \vec{u}_{i-1,k}) + p_{\max} 1_N \\
1_N \otimes \begin{bmatrix}0\\1\\0\end{bmatrix} \Lambda x_{i,k} - v_{\min} 1_N \\
-1_N \otimes \begin{bmatrix}0\\1\\0\end{bmatrix} \Lambda x_{i,k} + v_{\max} 1_N \\
1_N \otimes \begin{bmatrix}0\\0\\1\end{bmatrix} \Lambda x_{i,k} - a_{\min} 1_N \\
-1_N \otimes \begin{bmatrix}0\\0\\1\end{bmatrix} \Lambda x_{i,k} + a_{\max} 1_N \\
-u_{\min} 1_N \\
u_{\max} 1_N
\end{bmatrix}
$$

Obtaining the optimal control action

To obtain the optimal control action $\vec{u}_k^\star$ we minimise $J(e_{i,k}, \vec{u}_{i,k})$ with respect to $\vec{u}_{i,k}$. In the above this was done by taking the partial derivative and evaluating at 0. However, this approach did not include the constraints, which now appear as a hyper-plane in the cost function [6].

We note in the cost function at time k the quadratic term associated with $e_{i,k}$ and $\vec{u}_{i-1,k}$ is constant with respect to $\vec{u}_{i,k}$. Thus in a minimisation routine this term has no impact. We could alternatively write the cost function as

$$
J(e_{i,k}, \vec{u}_{i,k}) = \vec{u}_{i,k}^\mathsf{T}(\Phi^\mathsf{T}\bar{Q}\Phi + \bar{R})\vec{u} + 2\vec{u}^\mathsf{T}(-\Phi)^\mathsf{T}\bar{Q}(\Lambda e_{i,k} + \Phi\vec{u}_{i-1,k} - \vec{e}^\star). \tag{9}
$$

This is in the form of a convex function, where we could write

$$
J(\vec{u}_{i,k}) = \vec{u}_{i,k}^\mathsf{T} H \vec{u}_{i,k} + \vec{u}^\mathsf{T} f.
$$

Many convex optimisation solvers, including quadratic program solvers, permit the minimisation of the function with respect to the free variable subject to constraints:

$$\min_{\vec{u}_{i,k}} J(\vec{u}_{i,k})$$
$$\text{such that} \quad A_{in}\vec{u}_{i,k} \leq B_{in} \quad \text{and} \quad A_{eq}\vec{u}_{i,k} = B_{eq}$$

Numerical illustration

Let us quickly consider a numerical illustration of three vehicles. We consider a mechanical actuation lag of $\tau = 1.5$. We define our reference as $e^\star = [1, 0, 0]^\mathsf{T}$, and cost as $Q = \text{diag}\{100, 1, 1\}$ and $R = 10^{-3}$.

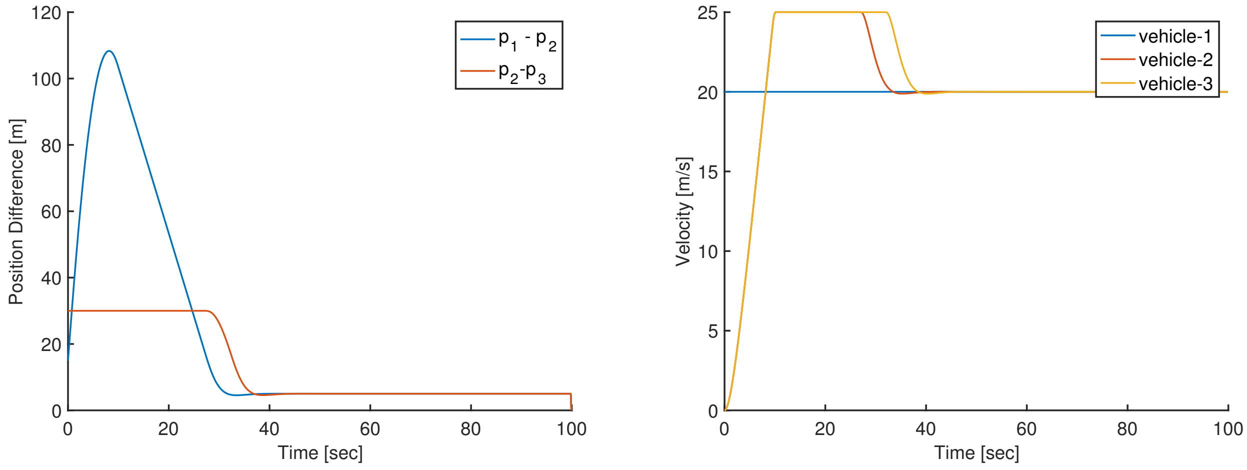

Figure 7: Numerical illustration of a platoon of three vehicles using MPC.

Figure 7 shows the position difference and velocity of three vehicles where the follower vehicles are using the constrained MPC. Note that the vehicles do not impact on the predecessor vehicle and the velocity remains constrained at the maximum velocity.

References

[1] K. J. Åström and B. Wittenmark, *Computer-Controlled Systems: Theory and Design.* Upper Saddle River, N.J.: Prentice Hall, Inc., 2011.

[2] E. F. Camacho and C. Bordons, *Model Predictive Control.* New York, N.Y.: Springer-Verlag, 1999.

[3] S. Feng, Y. Zhang, S. E. Li, Z. Cao, H. X. Liu, and L. Li, String stability for vehicular platoon control: Definitions and analysis methods, *Annual Reviews in Control*, vol. 47, pp. 8197, 2019.

[4] G. C. Goodwin, M. M. Serón, and J. A. De Doná, *Constrained Control & Estimation – An Optimization Perspective.* London: Springer Verlag, 2005.

[5] J. M. Kennedy, J. Heinovski, D. E. Quevedo, and F. Dressler, "Centralized Model Predictive Control with Human-Driver Interaction for Platooning," *IEEE Transactions on Vehicular Technology*, May. 2023

[6] J. M. Maciejowski, *Predictive Control with Constraints.* Prentice-Hall, 2002.

[7] D. E. Quevedo, R. Aguilera, and T. Geyer, "Model predictive control for power electronics applications," in *Handbook of Model Predictive Control*, pp. 551–580, Birkhäuser, Basel, 2019.

[8] D. E. Quevedo, H. Bölcskei, and G. C. Goodwin, "Quantization of filter bank frame expansions through moving horizon optimization," *IEEE Transactions of Signal Processing*, Feb. 2009.

[9] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design.* Madison, WI: Nob Hill Publishing, 2009.