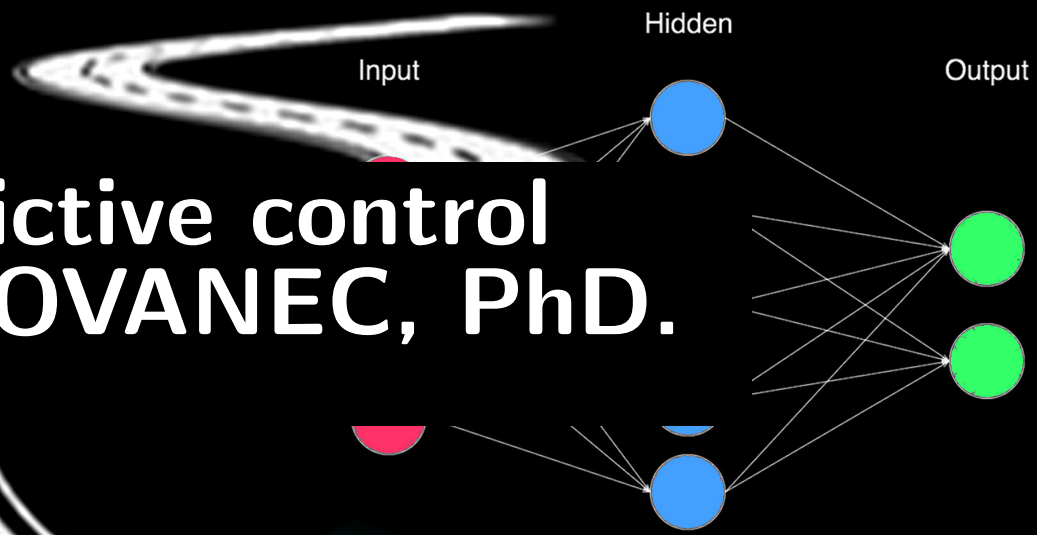
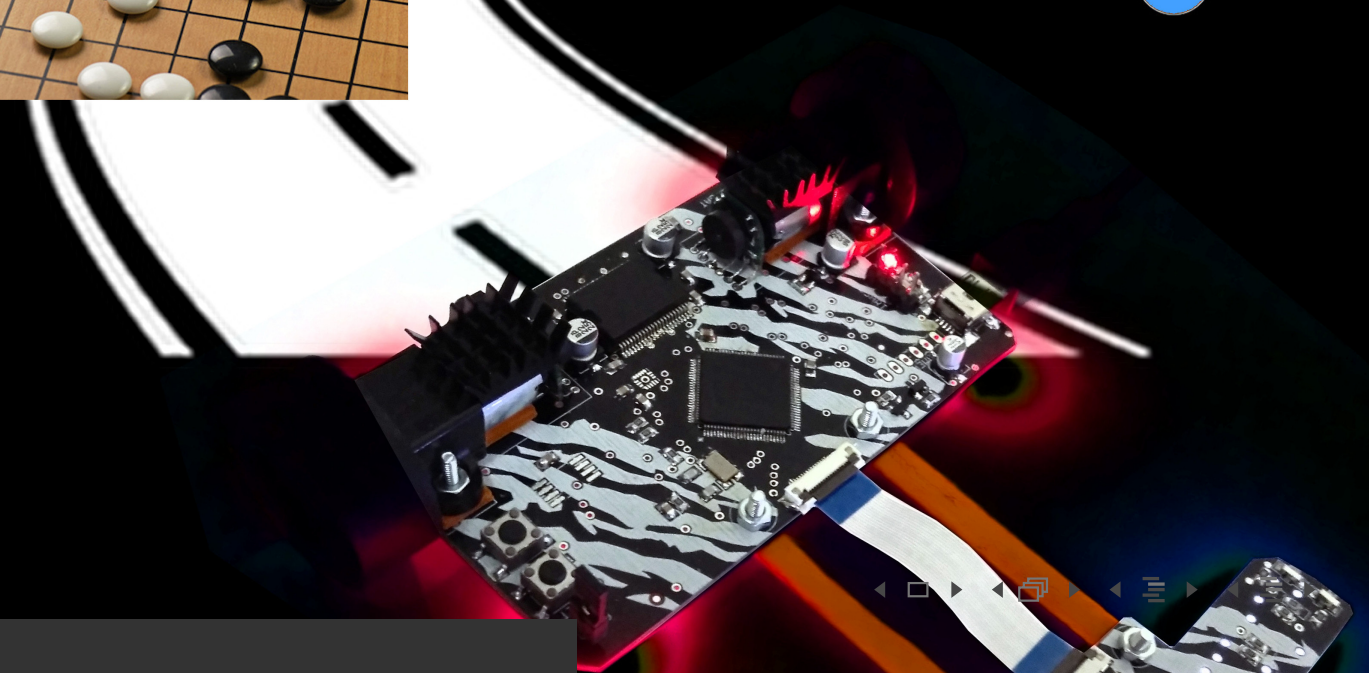
$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_{t+1} + \lambda \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right]$$

(The New Action Value = The Old Value) + The Learning Rate × (The New Information − the Old Information)

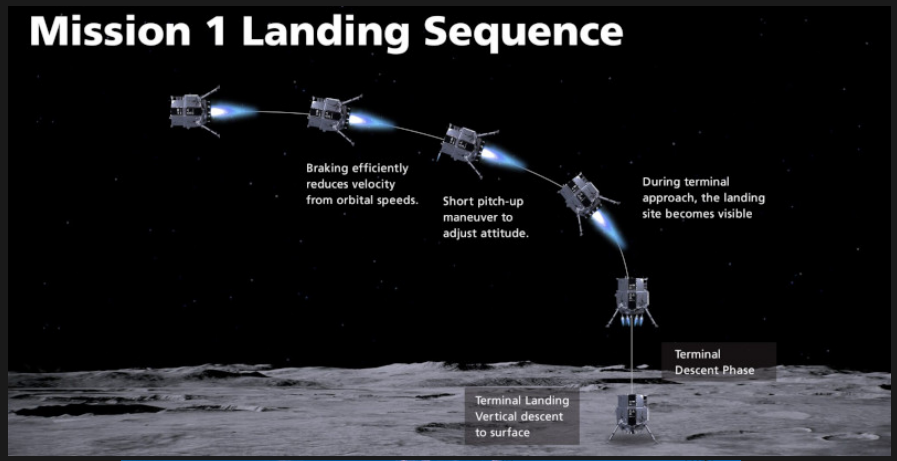


model predictive control

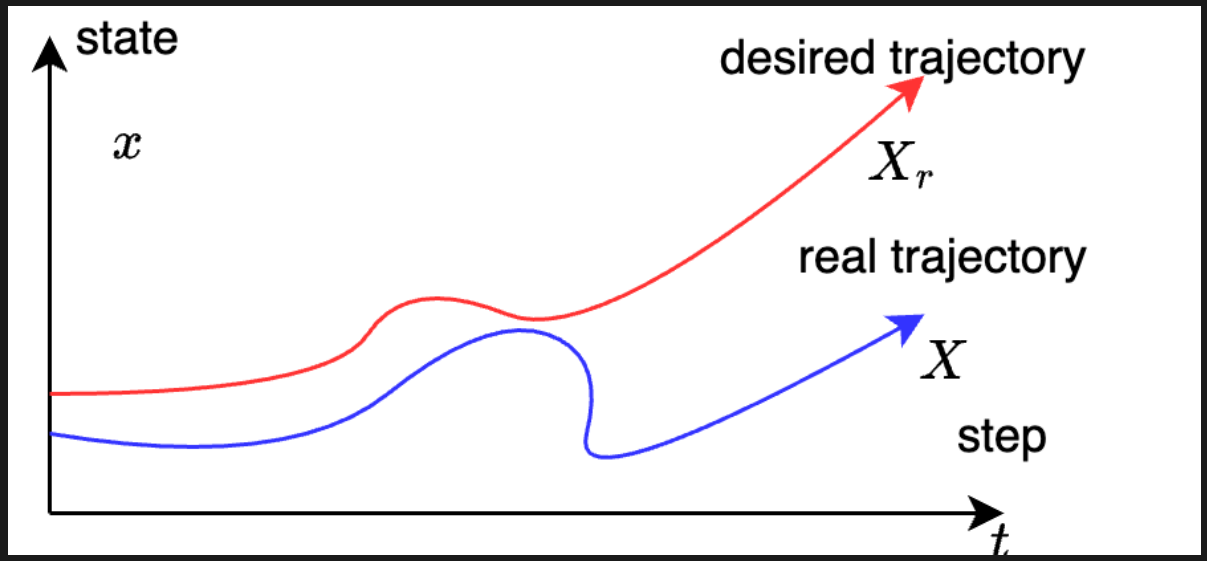
Michal CHOVANEC, PhD.



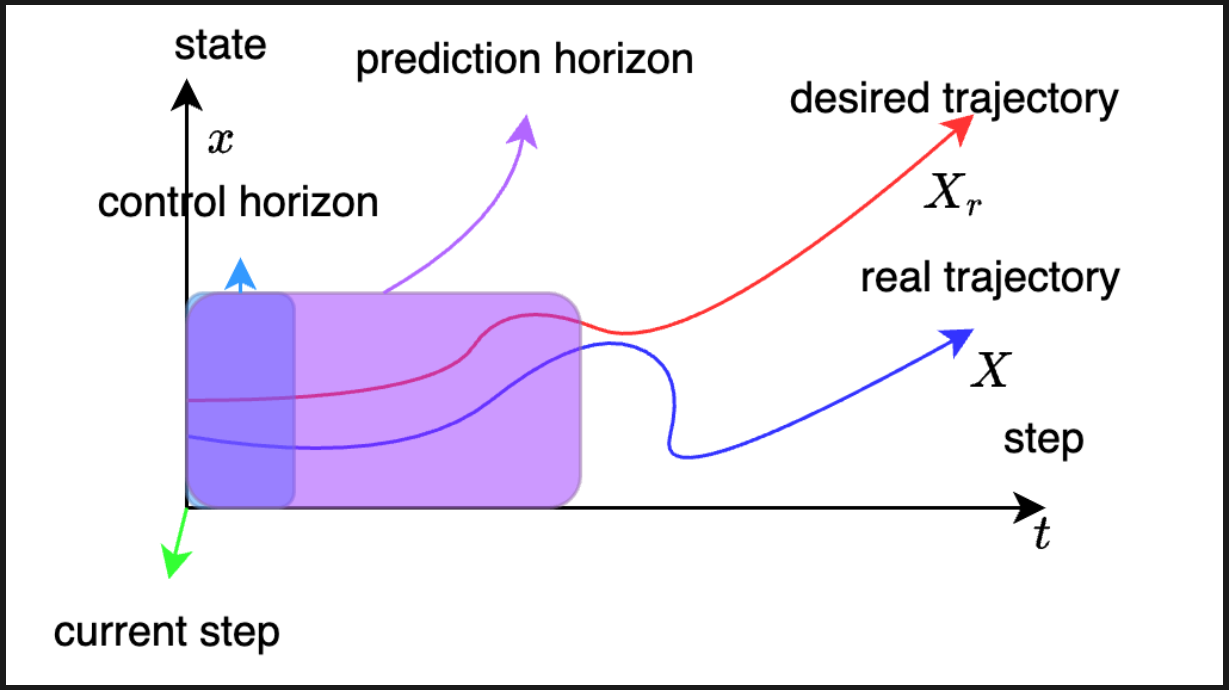
MPC - applications



MPC - overview



MPC - overview



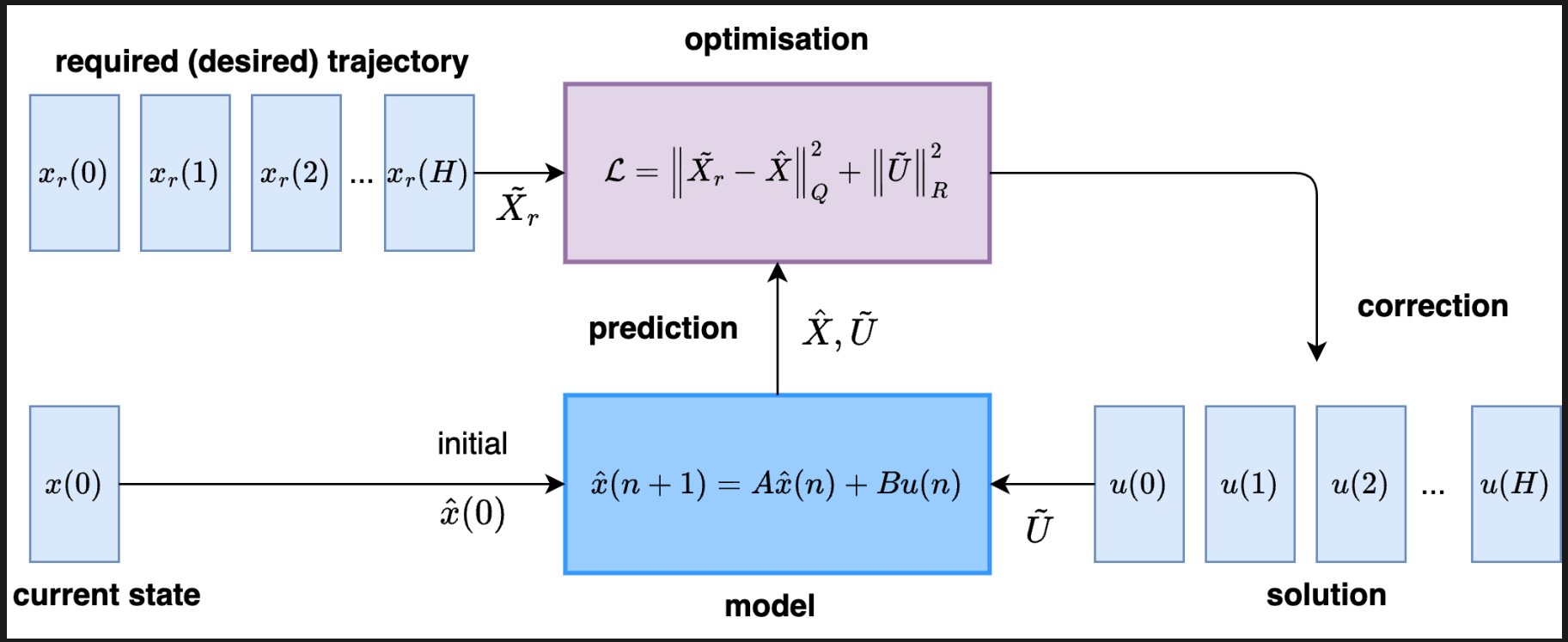
state vector,
e.g. (position, velocity, angle, angular rate)

$$x = \begin{bmatrix} x_0 \\ x_1 \\ .. \\ x_{n-1} \end{bmatrix}$$

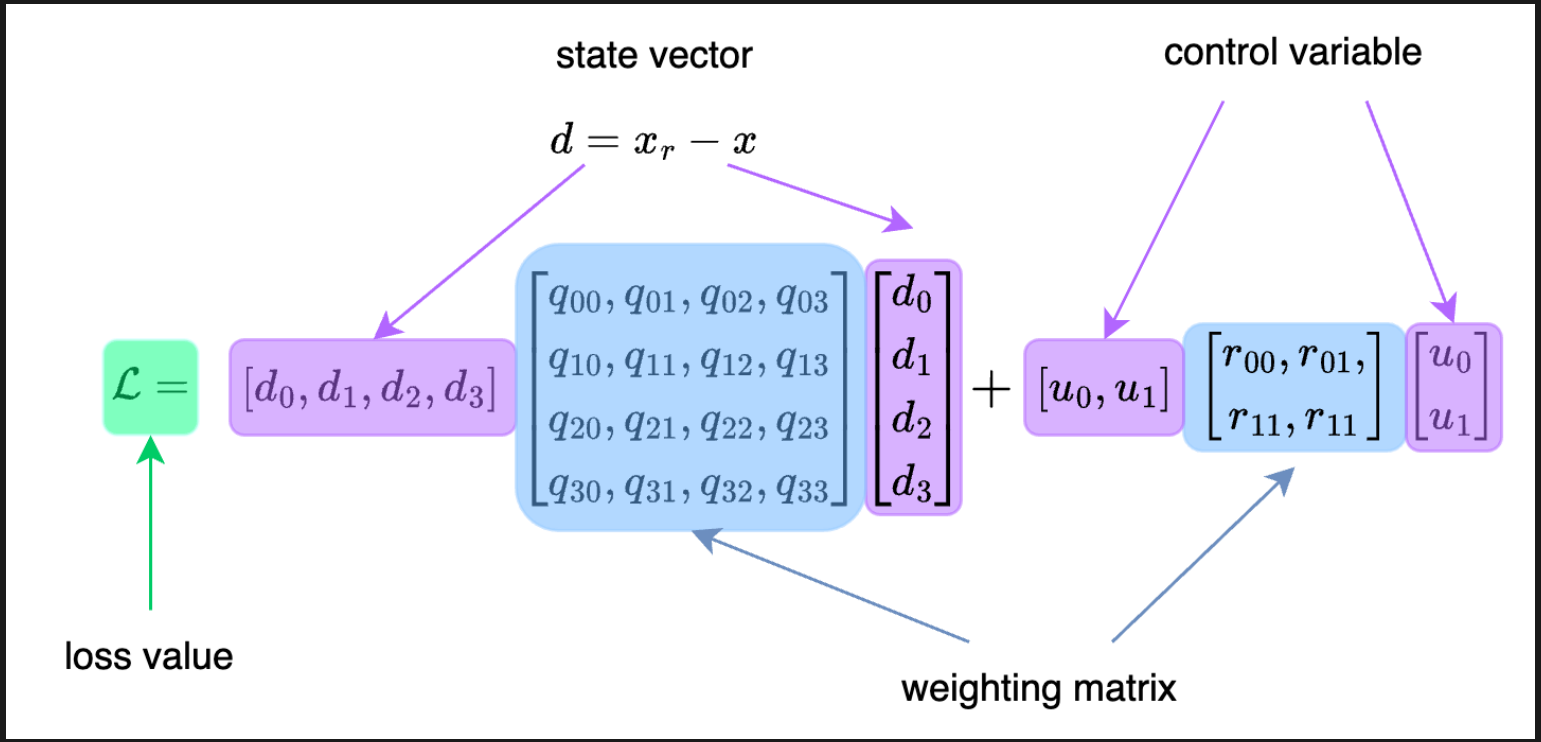
control variable,
e.g. (left speed, right speed)

$$u = \begin{bmatrix} u_0 \\ u_1 \\ .. \\ u_{m-1} \end{bmatrix}$$

MPC - overview



quadratic problem formulation



quadratic problem formulation

loss (cost) function is quadratic, with weighting terms Q and R,

$$\mathcal{L} = \sum_{h=0}^{H-1} \left((x_r(h) - x(h))^T Q (x_r(h) - x(h)) \right) + \Delta u^T(h) R \Delta u(h),$$

$$u(n) = u(n-1) + \Delta u(n),$$
$$\text{s.t.} \quad x(n+1) = Ax(n) + Bu(n).$$

where :

- H is prediction horizon steps
- A is matrix, $n \times n$
- B is matrix, $n \times m$
- Q is matrix, $n \times n$
- R is matrix, $m \times m$
- $_{\Delta}u$ is controller output
- where n is system orders, and m system inputs count

unrolling sequence

rewrite into form where initial conditions **depends only** on $x(n)$ and $u(n-1)$

$$
\begin{aligned}
x(n+1) &= Ax(n) + Bu(n) \\
&= Ax(n) + B(u(n-1) +_\Delta u(n)) \\
x(n+2) &= Ax(n+1) + B(u(n) +_\Delta u(n+1)) \\
&= A^2x(n) + (AB+B)u(n-1) + (AB+B)_\Delta u(n) + B_\Delta u(u) \\
x(n+3) &= Ax(n+2) + B(u(n+1) +_\Delta u(n+2)) \\
&= A^3x(n) + (A^2+AB+B)u(n-1) \\
&\quad + (A^2B+AB+B)_\Delta u(n) \\
&\quad + (AB+B)_\Delta u(n+1) + B_\Delta u(n+2) \\
&\quad \ldots
\end{aligned}
$$

matrix formulation

rewrite into matrix form

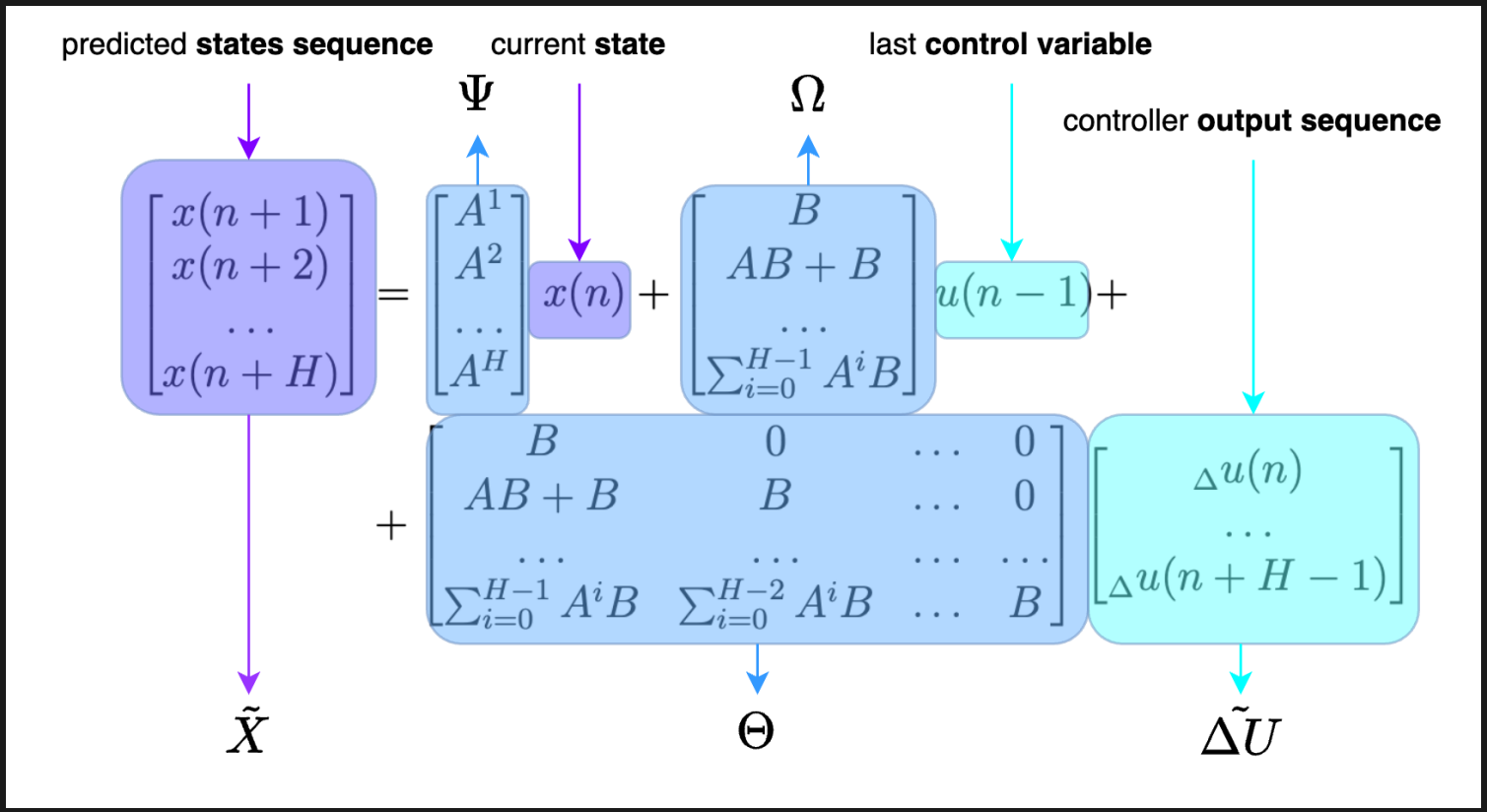
$$\begin{bmatrix} x(n+1) \\ x(n+2) \\ \dots \\ x(n+H) \end{bmatrix} = \begin{bmatrix} A^1 \\ A^2 \\ \dots \\ A^H \end{bmatrix} x(n) + \begin{bmatrix} B \\ AB \\ \dots \\ \sum_{i=0}^{H-1} A^i B \end{bmatrix} u(n-1) +$$

$$+ \begin{bmatrix} B & 0 & \dots & 0 \\ AB+B & B & \dots & 0 \\ \dots & \dots & \dots & \dots \\ \sum_{i=0}^{H-1} A^i B & \sum_{i=0}^{H-2} A^i B & \dots & B \end{bmatrix} \begin{bmatrix} \Delta u(n) \\ \dots \\ \Delta u(n+H-1) \end{bmatrix}$$

in compaxt form

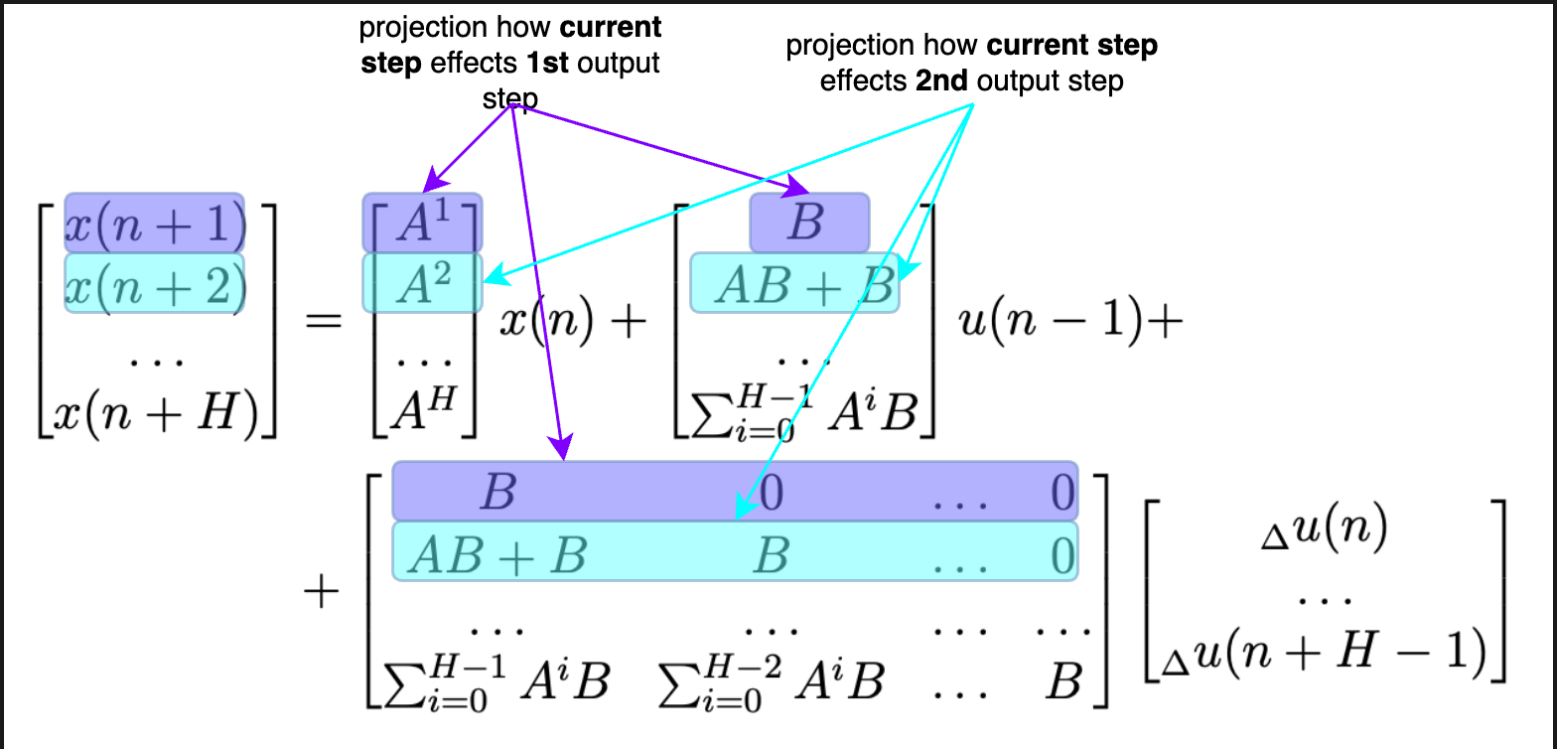
$$\tilde{X} = \Psi x(n) + \Omega u(n-1) + \Theta_{\Delta} \tilde{U}$$

matrix formulation

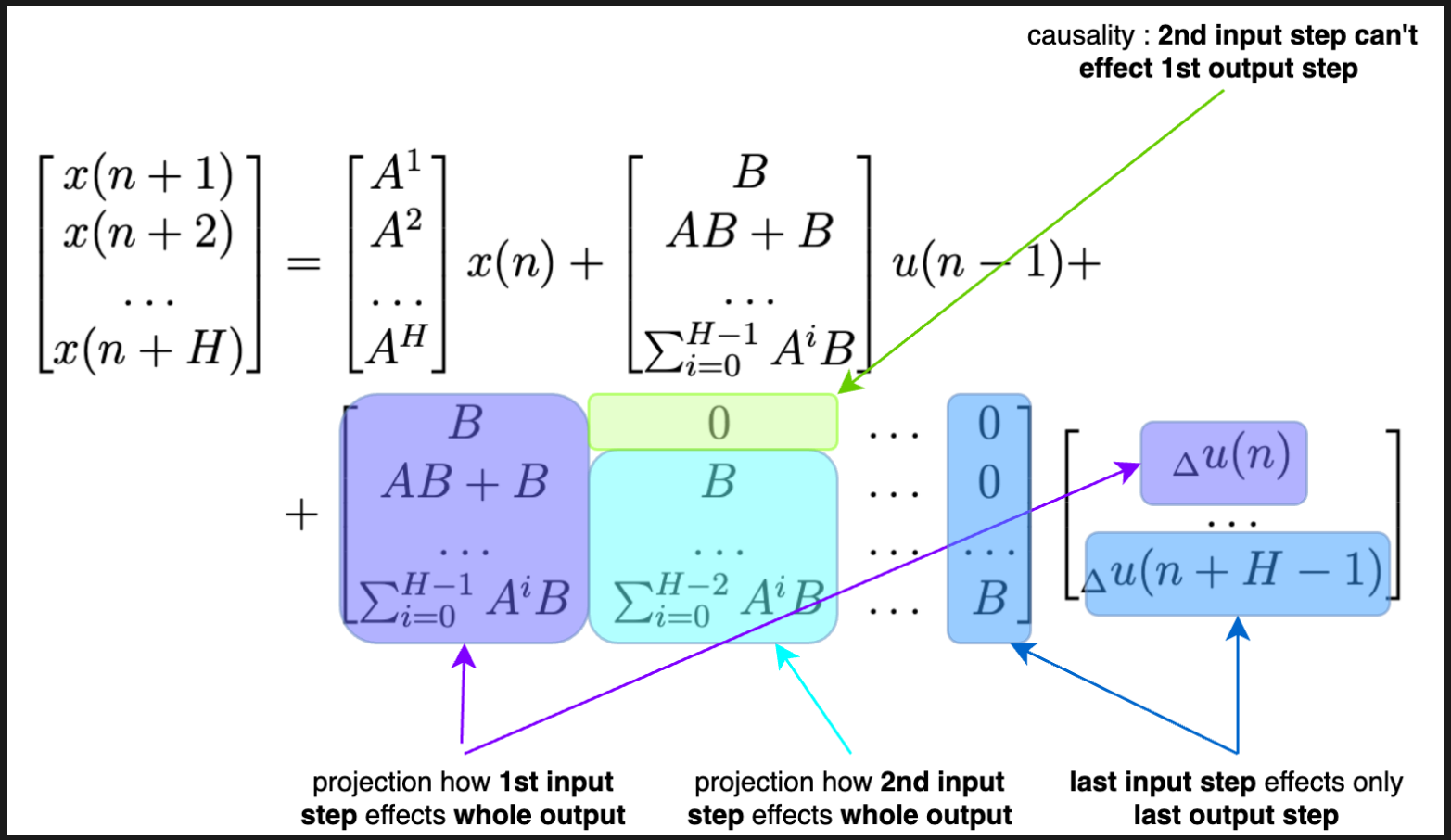


$$\tilde{X} = \Psi x(n) + \Omega u(n-1) + \Theta_{\Delta}\tilde{U}$$

n-th step input to output projection



projection of controll $\Delta u(n)$ to output sequence



put into quadratic loss (cost) function

$$\mathcal{L} = {}_{\Delta}\tilde{U}^{T}\tilde{R}_{\Delta}\tilde{U} + (\tilde{X}_r - \tilde{X})^{T}\tilde{Q}(\tilde{X}_r - \tilde{X})$$

after substitution

$$S = \tilde{X}_r - \Psi\tilde{X} - \Omega u(n-1)$$

we obtain

$$\begin{aligned}\mathcal{L} &= {}_{\Delta}\tilde{U}^{T}\tilde{R}_{\Delta}\tilde{U} + (S - \Theta_{\Delta}\tilde{U})^{T}\tilde{Q}(S - \Theta_{\Delta}\tilde{U}) \\ &= {}_{\Delta}\tilde{U}^{T}\tilde{R}_{\Delta}\tilde{U} + S^{T}\tilde{Q}S - S^{T}\tilde{Q}\Theta_{\Delta}\tilde{U} - {}_{\Delta}\tilde{U}^{T}\Theta^{T}\tilde{Q}S + {}_{\Delta}\tilde{U}^{T}\Theta^{T}\tilde{Q}\Theta_{\Delta}\tilde{U}\end{aligned}$$

finding minima

find derivative with respect to $_{\Delta}\tilde{U}$

$$\frac{\partial \mathcal{L}}{\partial_{\Delta}\tilde{U}}:$$

$$\frac{\partial_{\Delta}\tilde{U}^{T}\tilde{R}_{\Delta}\tilde{U}}{\partial_{\Delta}\tilde{U}} = 2\tilde{R}_{\Delta}\tilde{U}$$

$$\frac{\partial S^{T}\tilde{Q}S}{\partial_{\Delta}\tilde{U}} = 0$$

$$\frac{\partial -S^{T}\tilde{Q}\Theta_{\Delta}\tilde{U}}{\partial_{\Delta}\tilde{U}} = -\Theta^{T}\tilde{Q}S$$

$$\frac{\partial -_{\Delta}\tilde{U}^{T}\Theta^{T}\tilde{Q}S}{\partial_{\Delta}\tilde{U}} = -\Theta^{T}\tilde{Q}S$$

$$\frac{\partial_{\Delta}\tilde{U}^{T}\Theta^{T}\tilde{Q}\Theta_{\Delta}\tilde{U}}{\partial_{\Delta}\tilde{U}} = 2\Theta^{T}\tilde{Q}\Theta_{\Delta}\tilde{U}$$

finding minima

put derivate equal to zero, and solve

$$\frac{\partial \mathcal{L}}{\partial {}_{\Delta}\tilde{U}} = 2\tilde{R}\,{}_{\Delta}\tilde{U} - 2\Theta^T\tilde{Q}S + 2\Theta^T\tilde{Q}\Theta\,{}_{\Delta}\tilde{U}$$

$$0 = 2\tilde{R}\,{}_{\Delta}\tilde{U} - 2\Theta^T\tilde{Q}S + 2\Theta^T\tilde{Q}\Theta\,{}_{\Delta}\tilde{U}$$

$$(\tilde{R} + \Theta^T\tilde{Q}\Theta)\,{}_{\Delta}\tilde{U} = \Theta^T\tilde{Q}S$$

and obtain **analytical solution** for model predictive controll

$${}_{\Delta}\tilde{U} = (\tilde{R} + \Theta^T\tilde{Q}\Theta)^{-1}\Theta^T\tilde{Q}S$$

full algorithm

given matrices :

$$\tilde{Q}, \tilde{R}, \Theta, \Phi, \Omega$$

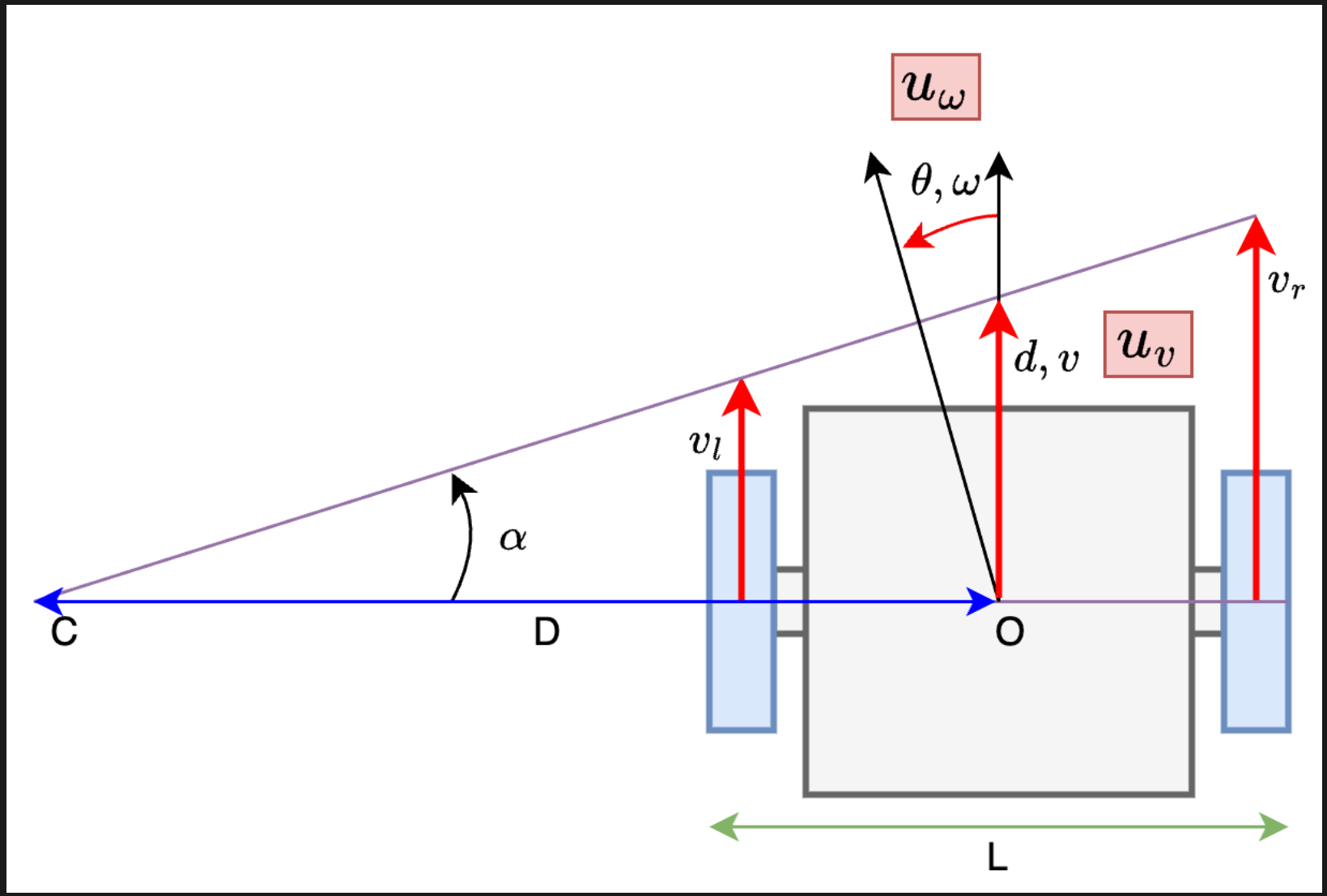
initialization (precompute) :

$$\Xi = (\tilde{R} + \Theta^T \tilde{Q} \Theta)^{-1} \Theta^T \tilde{Q}$$

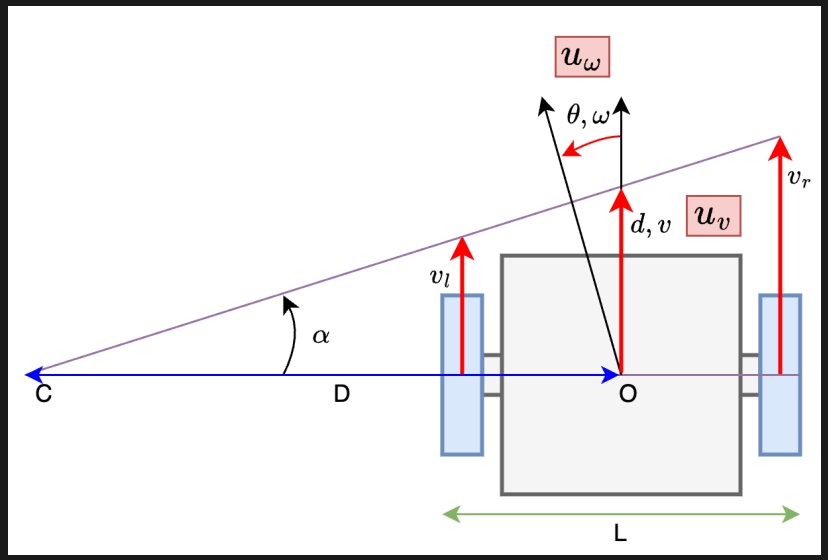
in loop :

$$\begin{aligned} E(n) &= X_r(n) - \Phi x(n) - \Omega u(n-1) \\ {}_{\Delta}u(n) &= \Xi E(n) \\ u(n) &= u(n-1) +_{\Delta} u(n) \end{aligned}$$

robot trajectory tracking example



robot trajectory tracking example



- state

$$x = \begin{bmatrix} v \\ d \\ \omega \\ \theta \end{bmatrix}$$

- control variable

$$\Delta u = \begin{bmatrix} \Delta u_v \\ \Delta u_\omega \end{bmatrix}$$

$$u(n) = u(n-1) + \Delta u(n)$$

state space model (continuos)

$$dx = Ax + Bu$$

$$A = \begin{pmatrix} -\frac{1}{\tau_{forward}} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{\tau_{turn}} & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad B = \begin{pmatrix} -\frac{k_{forward}}{\tau_{forward}} & 0 \\ 0 & 0 \\ 0 & -\frac{k_{turn}}{\tau_{turn}} \\ 0 & 0 \end{pmatrix}$$

- 4 poles : double in 0, $-\frac{1}{\tau_{forward}}$ and $-\frac{1}{\tau_{turn}}$
- 2 control inputs : forward velocity and turning velocity
- constrains for kick avoid (jerk) : $_{\Delta}u_{min} < {}_{\Delta}u < {}_{\Delta}u_{max}$
- constrains for motor limits : $u_{min} < u < u_{max}$

recommended reading