(F) Apr. 12 — Rover Kinematics

• $B \equiv$ body frame, $I \equiv$ inertial frame

$$X_I = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \Rightarrow \dot{X}_I = V_I = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix} \quad \omega = \omega_B$$

$$\dot{X}_B = V_B = \begin{bmatrix} V_{F,B} \\ V_{S,B} \\ \omega_B \end{bmatrix} = \overbrace{\begin{bmatrix} \cos(\theta(t)) & \sin(\theta(t)) & \emptyset \\ -\sin(\theta(t)) & \cos(\theta(t)) & \emptyset \\ \emptyset & \emptyset & 1 \end{bmatrix}}^{R} \overbrace{\dot{X}_I}^{V_I}$$

• wheel linear velocity: u_1, u_2, u_3

$$U = \begin{bmatrix} u_2 \\ u_3 \\ u_1 \end{bmatrix} = \begin{bmatrix} -\sin(\pi/3) & \cos(\pi/3) & d \\ \emptyset & -1 & d \\ \sin(\pi/3) & \cos(\pi/3) & d \end{bmatrix} \overbrace{\dot{X}_B}^{V_B} = \begin{bmatrix} R\omega_2 \\ R\omega_3 \\ R\omega_2 \end{bmatrix}$$

$[m]$ $\left[\frac{rad}{s}\right]$

$$\Rightarrow \dot{X}_B = \boxed{\begin{bmatrix} V_{F,B} \\ V_{s,B} \\ \omega_B \end{bmatrix} = \begin{bmatrix} (\sqrt{3}/3)(u_1 - u_2) \\ (1/3)(u_1 + u_2) - (2/3)u_3 \\ \left(\frac{1}{3d}\right)(u_1 + u_2 + u_3) \end{bmatrix}}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^T = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$$

$$\Rightarrow \dot{X}_I = \underbrace{\begin{bmatrix} \frac{\cos(\theta(t))}{(\cos(\theta(t))^2 + \sin(\theta(t))^2} & \frac{-\sin(\theta(t))}{(\;''\;)} & \emptyset \\ \frac{\sin(\theta(t))}{(\;''\;)} & \frac{\cos(\theta(t))}{(\;''\;)} & \emptyset \\ \emptyset & \emptyset & 1 \end{bmatrix}}_{M} \dot{X}_B$$

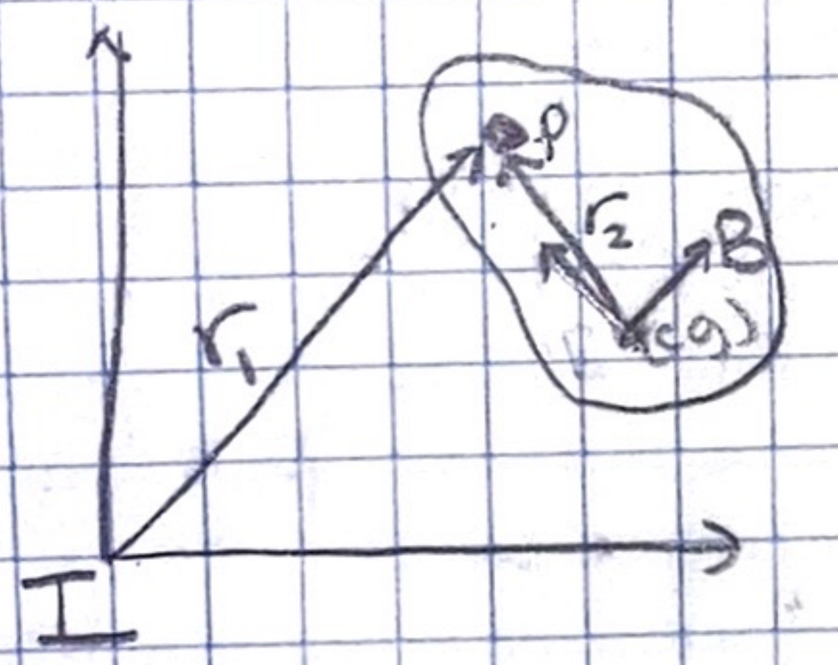
• Working w/ Mo:

For rover Kinematics:

- **Input**: control $[u_1; u_2; u_3]$ & $[x_n; y_n; \theta_n]$
- **Output**: vehicle motion $[x_{n+1}; y_{n+1}; \theta_{n+1}]$
- **Method**: (Forward) Euler Iteration

$$x_{n+1} = x_n + f(x, n)\,dt$$

• If using [Rigid Body Dynamics]



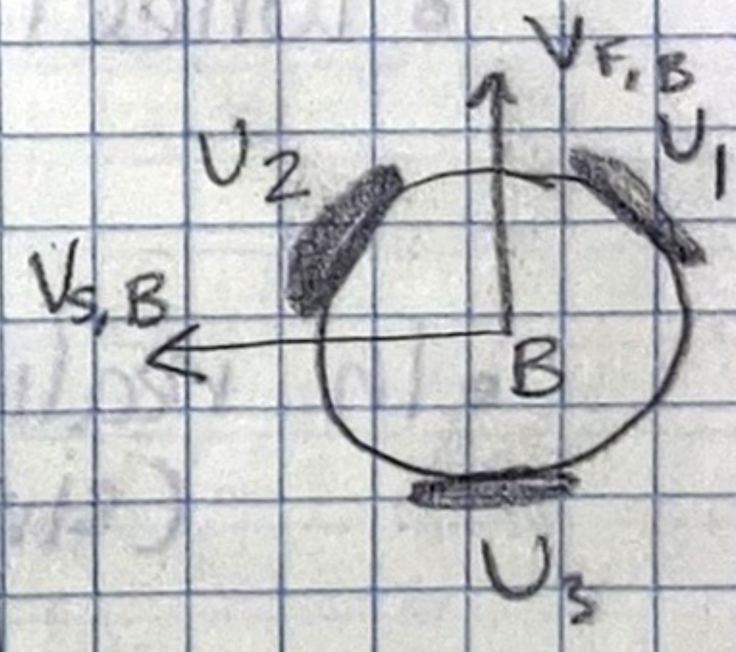
• At point P on rigid body:

$${}^I\frac{d\vec{r}}{dt} = {}^I\frac{d}{dt}(\vec{r}_1 + \vec{r}_2)$$
$$= {}^I\frac{d}{dt}\vec{r}_1 + {}^I\frac{d}{dt}\vec{r}_2$$
$$= {}^I\frac{d}{dt}\vec{r}_1 + {}^B\frac{d}{dt}\vec{r}_2 + {}^B\vec{\omega}^I \times \vec{r}_2$$

• If $\dot{r}_1 = V_I$, $\dot{r}_2 = V_B$:

Wrong (fix later)

$$\begin{bmatrix} V_{I,x} \\ V_{I,y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} V_{B,x} \\ V_{B,y} \end{bmatrix} + \left(\omega \times \begin{bmatrix} r_{I,x} \\ r_{I,y} \end{bmatrix}\right) \\ \omega \end{bmatrix}$$

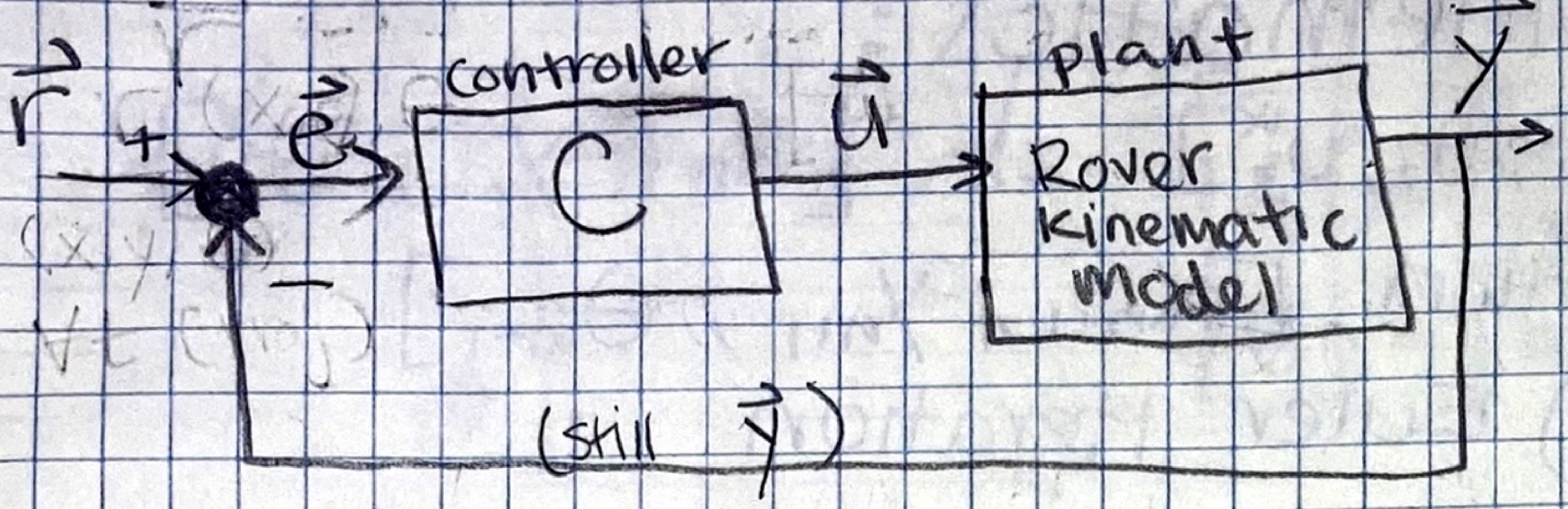


• **If using point mass** (points don't have ω):
↳ (This is what we will use b/c tracking CG)

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_I = \begin{bmatrix} \overbrace{\begin{bmatrix} \cos(\theta(t)) & -\sin(\theta(t)) \\ \sin(\theta(t)) & \cos(\theta(t)) \end{bmatrix}}^{\text{rotation matrix (instead of } \omega \text{ cross product)}} \begin{bmatrix} V_{F,B} \\ V_{S,B} \end{bmatrix} \\ \omega \end{bmatrix}$$

NOTE
- due to rolling constraint, can't spin wheels arbitrarily
  ↳ NEED TO DETERMINE CONSTRAINT
- in reality, doesn't rotate about CG...? Ask!

Moving forward:



- $\vec{r}$: desired reference signal/state
  - $X_I = \begin{bmatrix} x \\ y \\ \theta \\ \hline \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}$ desired

- $\vec{u}$: control command
  - wheel velocities, $\begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$
  - In reality, will be converted to PWM signal (simple unit conversion)

- $\vec{y}$: measured signal/state
  - X_I actual
  - subtracted from desired value $\vec{r}$ to get error $\vec{e} = (\vec{r} - \vec{y})$, which tells controller how to respond/correct accordingly.

Th Apr. 18 Designing Controller for Rovers

Notes • design continuous controller
- switch to ODE45
  - ↳ define $r(t)$ & call w/in derivs(t, x)

• derivs(t, x, u) ≡ keep for discrete controller
↳ control moved outside, given as input

$\boxed{x_p = X(i\_p)}$
$\boxed{x_{A_i} = X(i\_A_i)}$
$x_c = X(i\_c)$

$e = r(t) - C_p x_p$ (fxn)
$u_B = C_c x_c + D_c e$ (commanded x, y, θ)

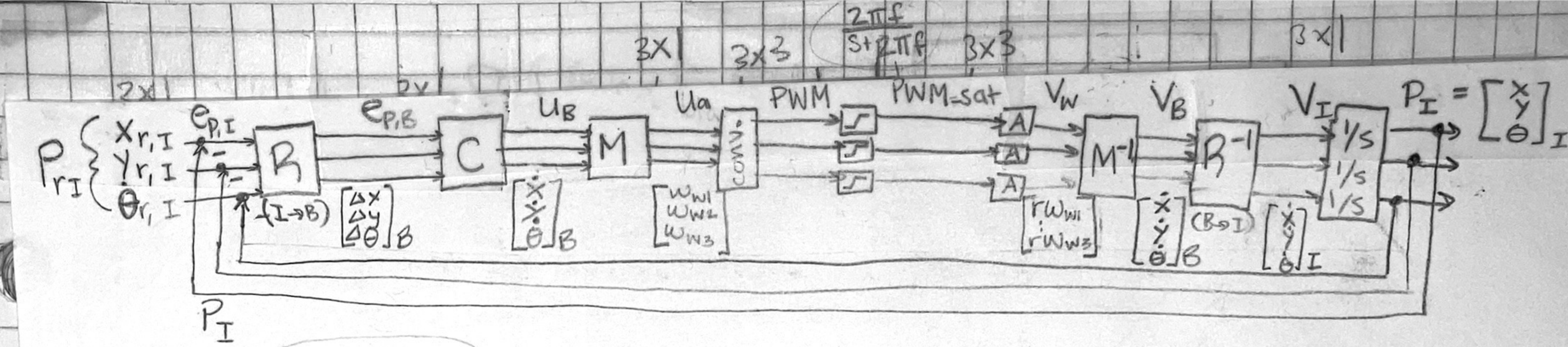
$y_{A_i} = C_A x_{A_i}$ (actuators)
$u_A = M u_B$
$\omega = M^{-1} y_A$

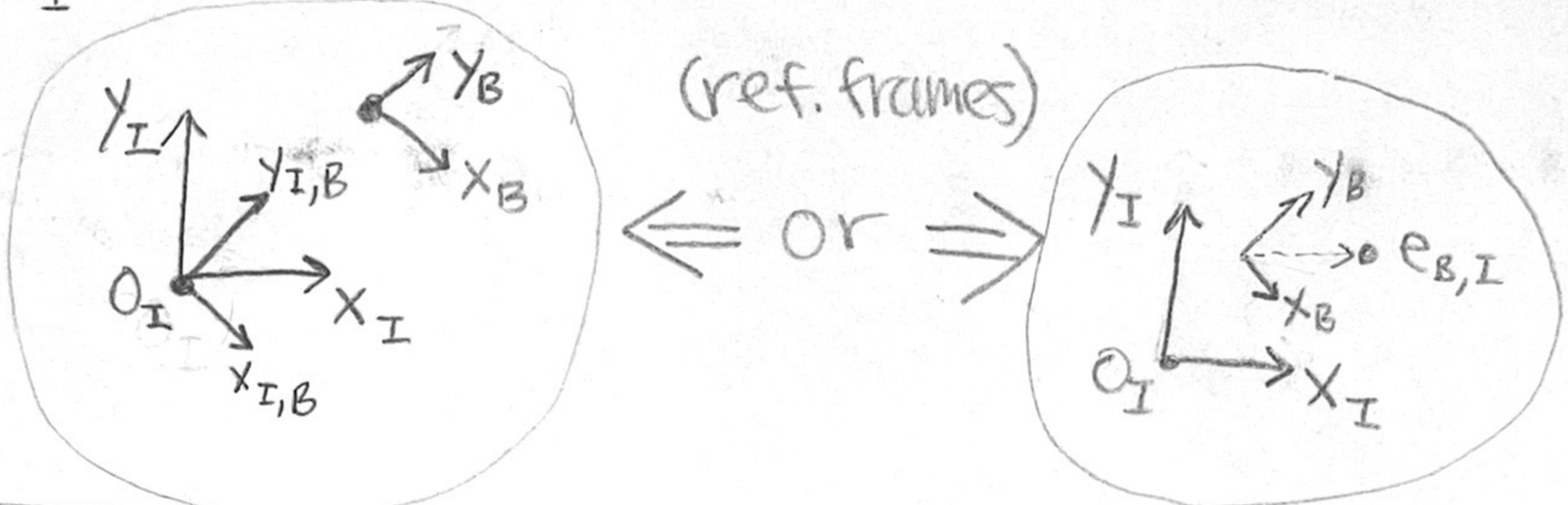
$\dot{x}_c = A_c x_c + B_c e$
$\dot{x}_{A_i} = A_A x_{A_i} + B_A u_A$ ~ sat(umin, u_{A_i}, umax)
$\dot{x}_p = f(x_p, \omega)$

$\dot{X} = [\dot{x}_c^T \ \dot{x}_{A_i}^T \ \dot{x}_p^T]^T$

Discrete Control

for k=1... $r(t_k)$
$e_k = r_k - y_k$
$(x_c)_{k+1} = (A_c)_D (x_c)_k + (B_c)_D e_k$
$u_k = (C_c)_D (x_c)_k + (D_c)_D e_k$ — matlab $\underbrace{\text{c2d}(u_{TF})}_{u_{D,TF}} \Rightarrow \text{ss}(u_{D,TF})$
[t, x] = ode45(@(t,x) derivs(t, x, u_k), [t_k, t_{k+1}], x_k, odeopt)
$x_{k+1} = X(\text{end}, :)^T$





Th Apr. 25

- M: converts ...
- M^{-1}: for sim, converts back to wheel speeds (modelling motion)
- Y_s: for sim, converts to $(x, y, \theta) = x_p$
- y: output

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

$$\underline{C(s)}: \quad \dot{x}_c = A_c x_c + B_c e \qquad \text{— } \underline{\text{control}}$$
$$u = C_c x_c + D_c e$$
$$A_c = \begin{bmatrix} A_{cx} & & 0 \\ & A_{cy} & \\ 0 & & A_{c\theta} \end{bmatrix}$$

$$\left\{\begin{aligned} \dot{x}_{Ai} &= A_A x_{Ai} + B_A u_{Ai} \\ y_{Ai} &= C_A x_{Ai} + D_A u_{Ai} \end{aligned}\right\} \quad i = \{1, 2, 3\} \qquad \text{— } \underline{\text{Actuators}}$$

$$\begin{bmatrix} u_{A_1} \\ u_{A_2} \\ u_{A_3} \end{bmatrix} = Mu, \quad \omega = M^{-1} \begin{bmatrix} y_{A_1} \\ y_{A_2} \\ y_{A_3} \end{bmatrix}$$

$$\left\{\begin{aligned} \dot{x}_p &= A_p x_p + B_p \omega_w \\ y &= C_p x_p + D_p \omega \end{aligned}\right.$$

- r, e: $\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_I \to \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_B \to C(s)$
- u_B: $\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_B \to M$
- $Mu_b = u_a$: $\begin{bmatrix} \omega_{w1} \\ \omega_{w2} \\ \omega_{w3} \end{bmatrix} \xrightarrow{\text{PWM conv.}} \boxed{\text{Actuator}}$

Actuator: $V_W \to M^{-1}$

- $M^{-1} y_a$: $V_B = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_B \to R^{-1} \to \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_I$

__Plant__: $\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_I \to \boxed{\frac{1}{s}} \to \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_B$

Th Apr. 25

- convert to PWM
- Saturate PWM

↓ for u_A

$V = R\omega$

- $e = r(t) - y$
- u : control DoF commands (x, y, θ)
- M: converts u to 3 wheel speeds $\rightarrow$ 3 PWM signals
- M^{-1} : for sim, converts back to wheel speeds (modelling motion)
- Y_s : for sim, converts to $(x, y, \theta) = x_p$
- y : output

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

$C(s)$: $\dot{x}_c = A_c x_c + B_c e$ — control

$u = C_c x_c + D_c e$

$$A_c = \begin{bmatrix} A_{cx} & & 0 \\ & A_{cy} & \\ 0 & & A_{c\theta} \end{bmatrix}$$

$$\left\{\begin{aligned} \dot{x}_{A_i} &= A_A x_{A_i} + B_A u_{A_i} \\ y_{A_i} &= C_A x_{A_i} + D_A u_{A_i} \end{aligned}\right\} \; i = \{1, 2, 3\}$$ — Actuators

$$\begin{bmatrix} u_{A_1} \\ u_{A_2} \\ u_{A_3} \end{bmatrix} = Mu, \quad \omega = M^{-1} \begin{bmatrix} y_{A_1} \\ y_{A_2} \\ y_{A_3} \end{bmatrix}$$

$$\left\{\begin{aligned} \dot{x}_p &= A_p x_p + B_p \omega_w \\ y &= C_p x_p + D_p \omega \end{aligned}\right.$$

- r, e: $\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_I \rightarrow \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_B \rightarrow C(s)$
- u_B: $\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_B \rightarrow M$
- $M u_b = u_a$: $\begin{bmatrix} \omega_{w1} \\ \omega_{w2} \\ \omega_{w3} \end{bmatrix} \xrightarrow{\text{PWM conv.}} \boxed{\text{Actuator}}$

Actuator: $V_W \longrightarrow M^{-1}$

- $M^{-1} y_a$: $V_B = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_B \rightarrow R^{-1} \rightarrow \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_I$

Plant: $\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}_I \rightarrow \boxed{\frac{1}{s}} \rightarrow \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}_B$

Th Apr. 25

☆
- convert to PWM
- saturate PWM

$\downarrow$ for u_A