Challenges in safety:
- Learning-based control; safe learning
- Safety of vision-based systems
- Human-robot interaction

Autonomous System (can use open-loop or closed-loop control)
- Observation/Perception: sensing the environment and system's own configuration
- Decision-making: planning the next action or sequence of actions to achieve a goal
- Action: taking actions that update the environment and/or system's own configuration

Cyber-Physical Systems
- Systems combining computer software and physical processes with highly coupled behavior

What is a **Safety-Critical** System?
- Any system there exists potential outcomes or failures modes that are **unacceptable**, typically due to injury, loss of life, or severe material damage.
- **Failure Set**: all system states that are unacceptable
- **Unsafe Set**: states from which it is impossible to avoid a failure set in the future.
- E.g. autonomous car, drone delivery

Main questions:
- How to formally define safety for different autonomous systems?
- Developing a set of common tools to analyze and assuring the system safety.
- Discussion and mitigation of unique challenges that ML and data has introduced towards robot safety.

- [ ] Basics of Dynamical System
- [ ] State-space representation
- [ ] Need for safety analysis

State-space representation

State: $x(t) \in \mathbb{R}^n$ (compactly written as x) x_t

"characteristics of interest"

Control / Action: $u(t) \in \mathbb{R}^m$ u_t

inputs that we can choose at each instance of time.

Output / Observation: $y(t) \in \mathbb{R}^l$ y_t

outputs that are measurable (typically through some sensors)

e.g. speed (by speedometer)

xy-position (GPS)

- For now, assume $y = x$ (rarely the case)

system dynamics: explain how the system state evolves over time

continuous time

$t \in \mathbb{R}$

$$\frac{dx(t)}{dt} = \dot{x}(t) = f_c(x(t), u(t))$$

Common in control theory and system theory and formal methods

discrete time

$t \in \mathbb{Z}$

$$x_{t+1} = f_D(x_t, u_t)$$

More common in RL and robotics because of the easy of computation and optimisation

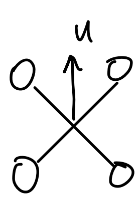
Common to obtain a discrete-time approximation from continuous-time dynamics.

$$x_{t+1} = \underbrace{x_t + \Delta T f_c(x_t, u_t)}_{f_D(x_t, u_t)}$$

← first order Euler approximation

only valid for small ΔT

example: longitudinal quadrotor motion



$$x = \begin{bmatrix} p_z \\ v_z \end{bmatrix}$$

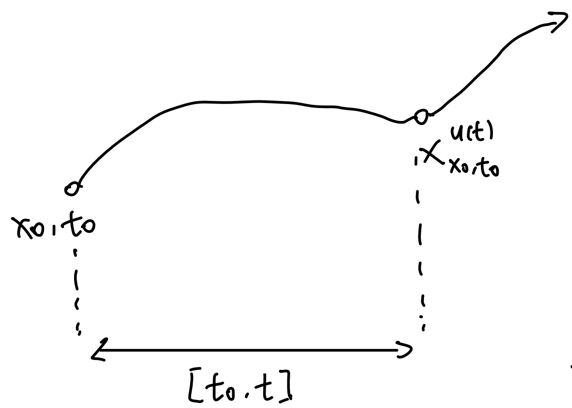
$u \rightarrow$ normalized thrust

$$\dot{x} = \begin{bmatrix} \dot{p}_z \\ \dot{v}_z \end{bmatrix} = \begin{bmatrix} v_z \\ g + k_0 u \end{bmatrix}$$

Trajectory notation: to make the time dependence of state explicit, we use

$$x = x(t) = x^{u}_{x_0, t_0}(t)$$

state arrived at time t starting from state x_0 at time t_0 and apply control $u(\cdot)$ over the time-interval $[t_0, t]$



$$x(t) = x_0 + v(t - t_0)$$

$x(\cdot)$

$u(\cdot)$