What is control theory?

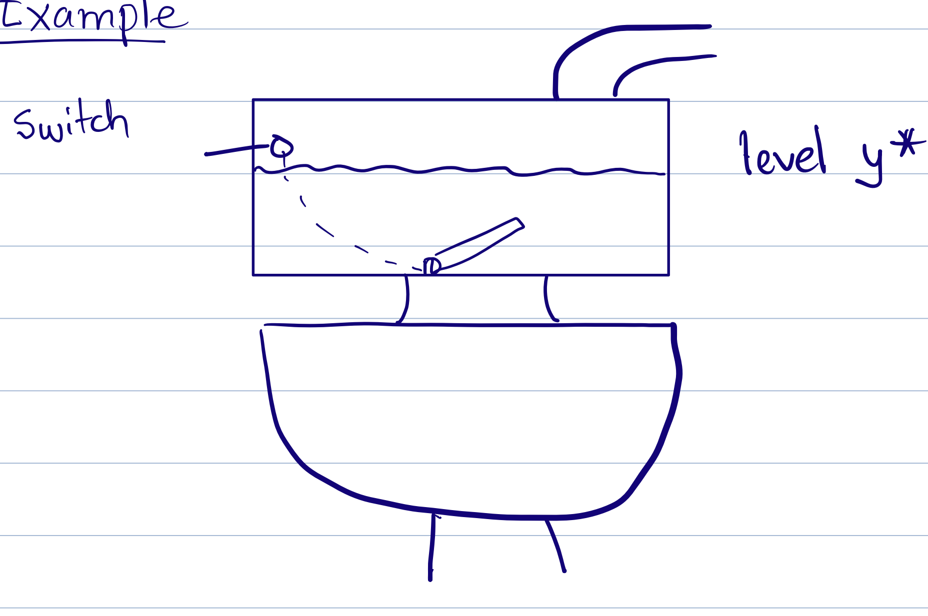
The art of making things do what you want them to do.

art : in this class, parameterized controllers or algorithms and ways of tuning those parameters

things : here : phenomena that can be represented with differential equations

what : in this class, follow some desired set-point, path, or trajectory

Example



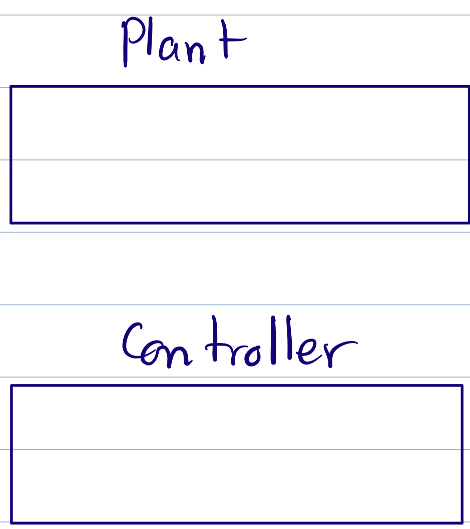
Road map

- Model ODE

- Control design
  - PID
  - State-feedback
  - Model Predictive control

- Requirements
  - Stability
  - Lyapunov theory

Discrete time model



Modeling, Differential equations

Continuous time version

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

$$\dot{x}(t) = f(x(t), u(t)) \quad \text{— } ①$$

$x(t) \in \mathbb{R}^n \qquad f: \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$

This could be written as

Short hand $\quad \dot{x} = f(x, u) \qquad x_{t+1} = f(x_t, g(x_t))$

If there is no input

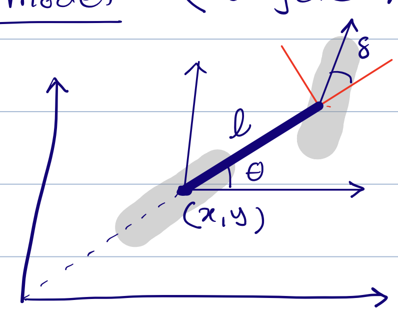
$\dot{x} = f(x)$ — also called autonomous system

e.g. $\quad u(t) = g(x(t))$

$$\dot{x} = f(x, u) = f(x, g(x)) = f'(x)$$

Example Vehicle model (bicycle / Kinematic vehicle)

State variables



Ref. Brian Paden et al.
Survey of motion planning and control for Self-driving urban 2016

front wheel steering angle δ (Control input)

$$\dot{x} = v \cos\theta$$

$$\dot{y} = v \sin\theta$$

$$\dot{\theta} = \frac{v}{l} \tan\delta$$

Def A solution of ① is any function

$x : \mathbb{R} \to \mathbb{R}^n$ such that $\dfrac{dx(t)}{dt} = f(x(t))$

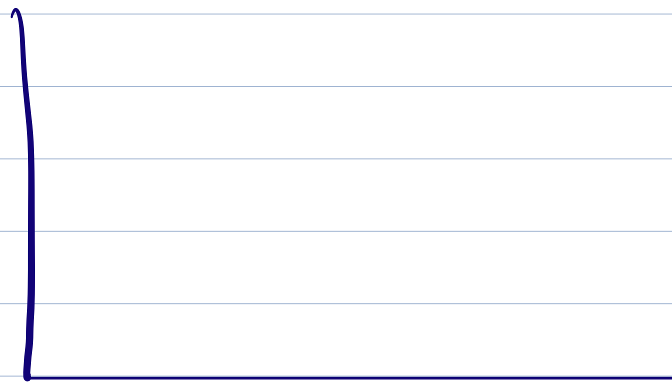
More generally for any input $u : \mathbb{R} \to \mathbb{R}^m$
$x : \mathbb{R} \to \mathbb{R}^n$ is a solution of ① if

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

Definition does not make sense if $u(t)$ is discontinuous.

We have to be careful about solutions

accel



When do they exist? When are they unique?

<u>Example</u> $\dot{x} = x^2$ with $x(0) = 1$

Example $\dot{x} = \sqrt{x}$ has two solutions

$$x(t) = 0$$

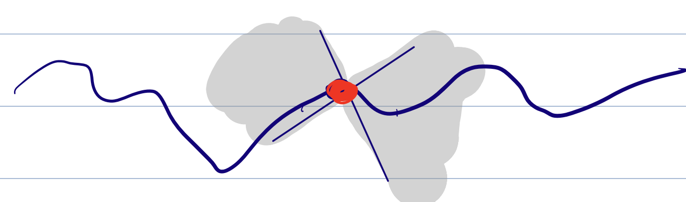
$$x(t) = t^{2/4}$$

Uniqueness is a problem

We will require additional condition on f.

Def $f : \mathbb{R}^n \to \mathbb{R}^n$ is Lipschitz continuous if $\exists L > 0$ such that for any pair $x, x' \in \mathbb{R}^n$

$$\| f(x) - f(x') \| \leq L \| x - x' \|$$



Example $6x + 4$; $|x|$ are Lipschitz

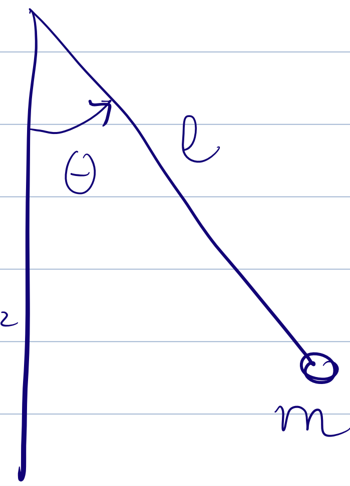
all differentiable functions with bounded derivatives are Lipschitz continuous

Non-Examples $\sqrt{x}$, x^2 are not Lipschitz

Thm. if $f(x(t), u(t))$ is Lipschitz continuous in the first argument and $u(\cdot)$ is

piece-wise continuous then
$\dot{x} = f(x(t), u(t))$ has unique solutions

Example Pendulum



$x_1 = \theta \quad x_2 = \dot{\theta}$

$\dot{x}_1 = x_2 \quad \dot{x}_2 = \frac{g}{l} \sin(x_1) - \frac{k}{m} x_2$

g : $9.8\ m/s^2$ on earth
m : mass

$$\begin{bmatrix} \dot{x}_2 \\ x_1 \end{bmatrix} = \begin{bmatrix} -g/l \sin(x_1) - \frac{k}{m} x_2 \\ x_2 \end{bmatrix}$$

When does the pendulum not move?

$\dot{x} = f(x) = 0$ set RHS $= 0$

$x_2 = 0 \quad x_1 = 0, \pi$

Def
A state $x^* \in \mathbb{R}^n$ is an equilibrium of (1)
if $f(x^*) = 0$.

Equilibria correspond to the steady state behavior of the system. $\lim_{t \to \infty} x(t)$

Transient behavior is what comes before steady state $x(t)$ $t < \infty$

Recall model - reality gap

As in automata models the gap exists here

tradeoff

more detailed Complex, intractable
more accurate harder to analyze

In this class we will focus on Linear ODEs

$$\dot{x} = f(x, u) = A(t)\,x(t) + B(t)\,u(t)$$

Any linear function can be represented in this form. $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$

$A(t) \in \mathbb{R}^{n \times n}$ a matrix; the entries

$B(t) \in \mathbb{R}^{m \times n}$ can be functions of time t

Linear time varying system (LTV)

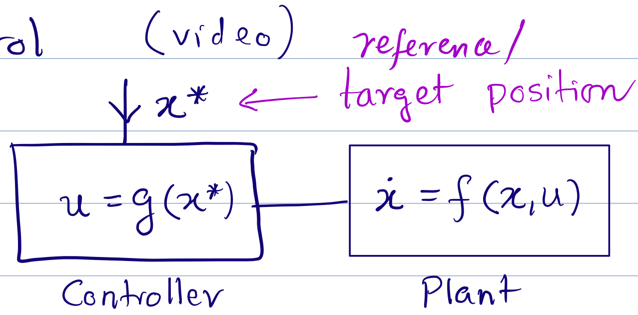
if $A(t)$ $B(t)$ are independent of time

then Linear time Invariant (LTI) System

Control Design

Simple strategy

Open loop control (video)



Does not use the state of the plant (no sensors)

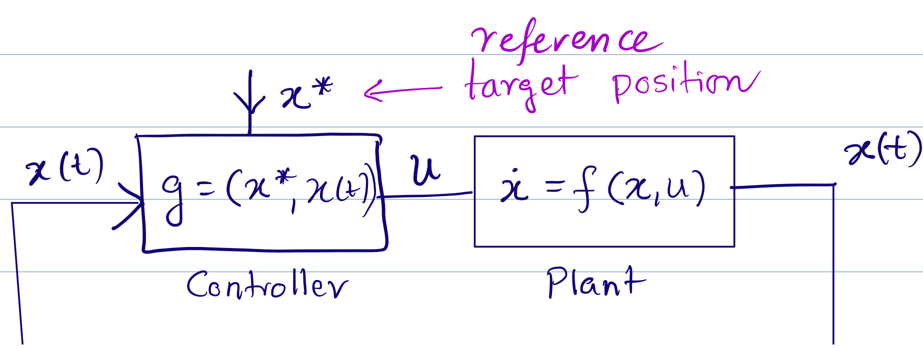
Used in Dryer, Coffee machines, Volume Control in audio
Emergency stop in vehicle, Sprinkler system

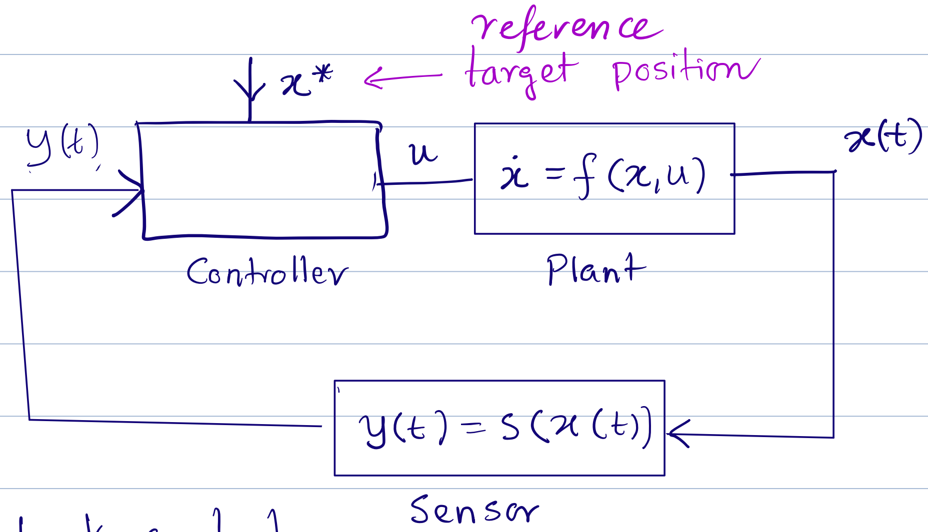
input $u(t)$ fixed as a function of the target x^*
Does not respond to state / environment
<u>Feedback</u> gives us better control

Closed-loop / Feedback control

$$u(t) = g(x^*, x(t))$$

e.g. $u(t) = g(\underbrace{x^* - x(t)}_{\text{error}})$





Error feedback control

Define the feedback $g(\,)$ as a scaled version of the error

$$u(t) = g(x^*, x(t)) = K\,\underbrace{(x^* - x(t))}_{\text{error}}$$

Proportional gain

Intuition: if car is really far to the left of the lane then we would move hard to the right; if it is slightly to the left, the we'd want to move to the right less aggresively.

This is also called <u>proportional control</u> (P-Control)

We can get more sophisticated

- g depends on the prediction of error (derivative)

∘ g depends on the history of error

PID

$u(t) =$

Tuning the gains is an art

<u>Example</u> $\dot{y}(t) = u(t) + d(t)$ ← disturbance

↑ Control input

Using only proportional control

We can make the steady state error small ($\uparrow K_p$)

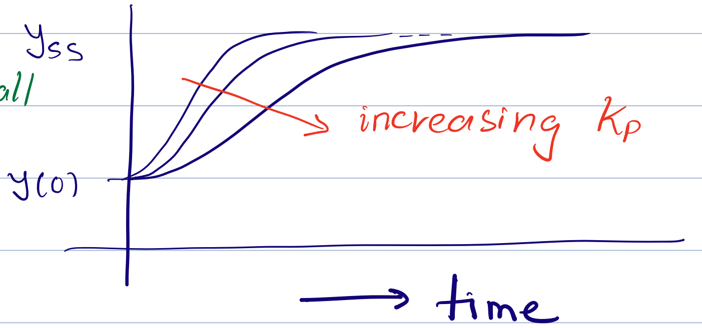
What is the transient behaviour of the system

Aside

What is solution of

$\dot{x} = -ax \qquad x(t) = x(0)e^{-at}$

We can make the steady state errors small at the expense of slower convergence / longer transients



Summary

ODEs language for studying Control

Solutions, Lipschitz Continuity existence, Uniqueness

Equilibria, Steady state, transient

Control design

open loop, feedback control

PID design

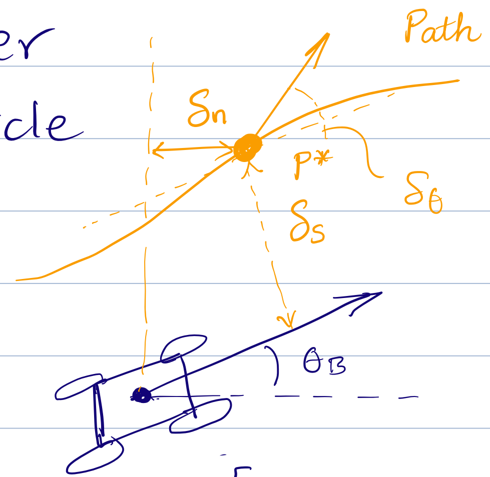
State Feedback Control for Linear Systems

Path following controller

Recall bicycle model for vehicle

Consider the state of the vehicle $[x_B, y_B, \theta_B\ v_B] \in \mathbb{R}^4$

Consider a target position p^* on a path (Chosen by a higher level planner)



$P_B = [x_B, y_B, \theta_B, v_B]$

The error is now a vector

$$e(t) = [\underbrace{\delta_s(t), \delta_n(t)}, \delta_\theta(t), \delta_v(t)]$$

along track error and cross-track error

Distance ahead or behind the target p^* in the instantaneous direction of motion

$$\delta_s = \cos\theta_B(t) \cdot (x^*(t) - x_B(t)) + \sin\theta_B(t)(y^*(t) - y_B(t))$$

Cross track error : orthogonal to the intended direction of motion

$$\delta_n = -\sin(\theta_B(t))(x^*(t) - x_B(t)) + \cos\theta_B(t)(y^*(t) - y_B(t))$$

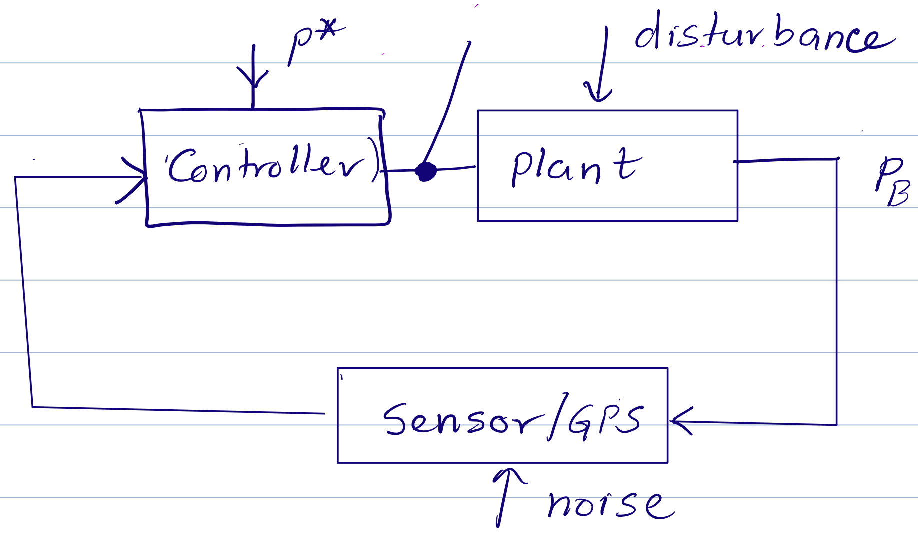
Heading error $\delta_\theta = \theta^*(t) - \theta_B(t)$

Velocity error $\delta_v = v(t) - v_B(t)$

a: throttle

δ: steering

$u(t) = [a(t), \delta(t)]$



Now you can apply PID or state-feedback control after linearization

Pure-pursuit controller

$$u(t) = K \begin{bmatrix} \delta_s \\ \delta_n \\ \delta_\theta \\ \delta_v \end{bmatrix} \qquad K = \begin{bmatrix} k_s & 0 & 0 & k_v \\ 0 & k_n & k_\theta & 0 \end{bmatrix}$$

This performs PD-Control to correct against along-track error and cross-track error

Stability