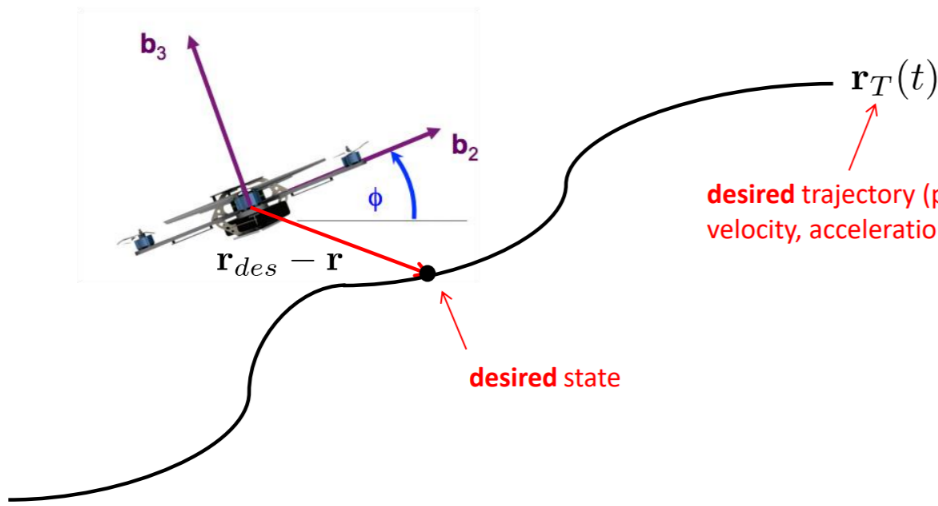
EECS C106B/206B Robotic Manipulation and Interaction **Spring 2021**

Lecture 2: (Feedback Control)

Scribes: Ashwin Vangipuram, Winnie Zhang

Contents

2.1 Trajectory Controller



2.1.1 Equilibrium

An equilibrium is a point in the state space where all the state variables remain constant as time progresses. It is often denoted as x_e.

2.1.2 Stability

Intuitively, stability is always defined relatively around the equilibrium; the controller should return to the equilibrium after it is pushed around or displaced from the initial point. The next section will discuss the various nuances of stability.

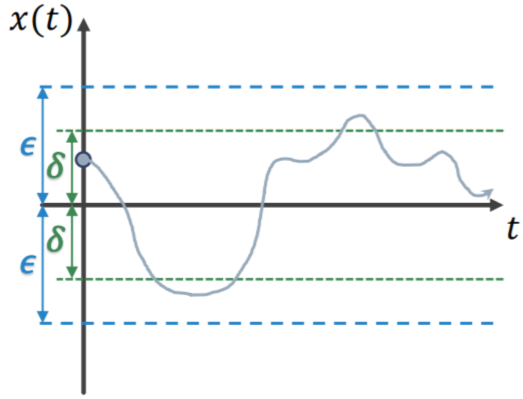
Fun fact: Galileo wanted to find out if the solar system is stable as he was proving the fact that the planets revolved around the Sun, rather than the Earth. While the answer may seem to be yes, we actually do not know the answer to this problem, and the question is now known to be the n-body problem. In the late 19th century, King Oscar II of Sweden established a prize for anyone that could solve this problem. Poincaré ended up winning the prize, and a magazine with his proof was to be published. However, one of the proofreaders for the article found an error in the proof. The error, in fact, was a relatively simple one; Poincaré assumed that any series would converge. He changed the resulting answer from the proof and used to prize money to reprint all the copies of the magazine.

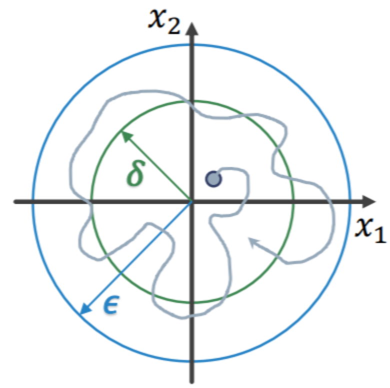
This story teaches us that we should check over people's proofs. The person who found the error in Poincaré's proof was a student similar to us!

2.2 Stability

2.2.1 Lyapunov Stability

An equilibrium point x_e of the system $\dot{x} = f(x)$ is called **Lyapunov stable** if for any $\epsilon > 0$, there exists a value $\delta(t_0, \epsilon) > 0$ such that if $||x(t_0, x_0) - x_e|| < \delta(t_0, \epsilon)$, then $||x(t, x_0) - x_e|| < \epsilon$ for all $t \geq t_0$.





- An equilibrium point is unstable if it is not Lyapunov stable.
- An equilibrium point is uniformly Lyapunov stable if $\delta = \delta(\epsilon)$ is constant for all ϵ.

Note that Lyapunov stability is a weak notion of stability. It just means that the system will remain close to the equilibrium point if it starts off near the equilibrium. However, it doesn't tell us if the system will ever reach the equilibrium point or if it'll stay at that equilibrium point for all future times.

2.2.2 Asymptotic Stability

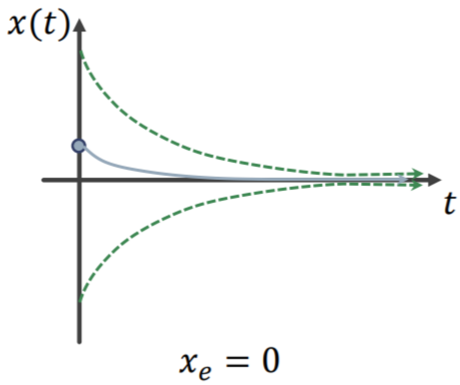
An equilibrium point is **asymptotically stable** if it is Lyapunov stable and additionally convergent:

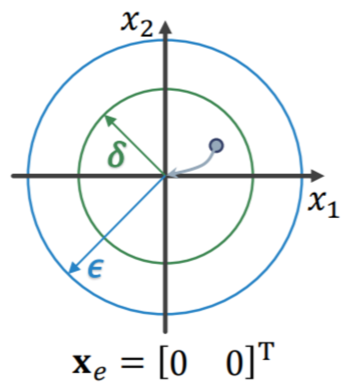
1. Lyapunov stable: for any $\epsilon > 0$, there exists a value $\delta(t_0, \epsilon) > 0$ such that if $||x(t_0, x_0) - x_e|| < \delta(t_0, \epsilon)$, then $||x(t, x_0)|| < \epsilon$ for all $t \geq t_0$.
2. Convergent: $x(t, x_0) \to x_e$ as t $\to \infty$

This still does not answer how fast the system will converge.

2.2.3 Exponential Stability

An equilibrium point $x_e = 0$ is **exponentially stable** if there exists coefficient $m \geq 0$ and rate $\alpha \geq 0$ such that $||x(t)|| \geq ||x_0||me^{-\alpha(t-t_0)}$ for all $t \geq t_0$ and for all x_0 in some ball around $x_e = 0$.





Note that exponential stability implies asymptotic stability by definition.

2.2.4 Local vs. Global

The system is locally stable if it is stable at an initial condition x_0. We were free to choose small δ in order to start x_0 near x_e.

The system is globally stable if it is stable for all initial conditions x_0.

2.2.5 Stability of LTI Systems

Consider a linear time-invariant (LTI) system:

$$\dot{x} = Ax, x \in \mathbb{R}^n, A \in \mathbb{R}^{n \times n}$$

- An LTI system is asymptotically stable if and only if all the eigenvalues of A have strictly negative parts.
- In LTI systems, asymptotically stability $\iff$ exponential stability.
- Out of scope: An LTI system is marginally stable if and only if all the eigenvalues of A have non-positive real parts, at least one has zero real part, and every eigenvalue with zero real parts has its algebraic multiplicity equal to it's geometry multiplicity. An equivalent statement is that all Jordan blocks corresponding to eigenvalue 0 must have size 1.

2.3 Control of a First Order System

Problem:

- State x and input u
- Kinematic model $\dot{x} = u$ (velocity)
- Want to follow trajectory $x^{des}(t)$

Strategy:

- Define error $e(t) = x^{des}(t) - x(t)$
- We want to find u such that $\dot{e} + K_p e = 0$
- This is since if $K_p > 0$, then $e(t) = e^{-K_p(t-t_0)}e(t_0)$ which shows that our error exponentially decays to 0.
- This u is $u(t) = \dot{x}^{des}(t) + K_p e(t)$ which we can check with

$$\begin{aligned}\dot{e} + K_p e &= \dot{x}^{des} - \dot{x} + K_p e \\ &= \dot{x}^{des} - u + K_p e \\ &= \dot{x}^{des} - \dot{x}^{des} - K_p e + K_p e = 0\end{aligned}$$

- This is called a P controller

2.4 Control of a Second Order System

Problem

- State x and input u
- Kinematic model $\ddot{x} = u$ (acceleration)
- Want to follow trajectory $x^{des}(t)$

Strategy

- Define error $e(t) = x^{des}(t) - x(t)$
- We want to find u such that $\ddot{e} + K_d\dot{e} + K_p e = 0$
- This is since when $K_p, K_d > 0$, the error again decays exponentially to 0
- The correct u here is $u(t) = \ddot{x}^{des}(t) + K_d\dot{e}(t) + K_p e(t)$ which can be checked with

$$\begin{aligned}\ddot{e} &= \ddot{x}^{des} - \ddot{x} \\ &= \ddot{x}^{des} - \ddot{x}^{des} - K_d\dot{e} - K_p e \\ &= -K_d\dot{e} - K_p e\end{aligned}$$

- This is called PD control

2.5 Control Gain Tuning

2.5.1 PD Control

- $u(t) = \ddot{x}^{des}(t) + K_d\dot{e}(t) + K_p e(t)$
- Proportional term (K_p) has a string (capacitance) response
- Derivative Term (K_d) has a dashpot (resistance) response

2.5.2 PID Control

- $u(t) = \ddot{x}^{des}(t) + K_d\dot{e}(t) + K_p e(t) + K_I \int_0^t e(\tau)\,\mathrm{d}x$
- Integral term (K_I) makes the steady state error go to 0. This term accounts for model error or disturbances
- You must be careful to reset the integral term to 0 when it gets too high and "winds up"
- PID control generates a third-order closed-loop system

2.5.3 Manual Tuning

General guidelines to consider when tuning a PID. See this Wikipedia link for more detail.

Parameter Increased	K_p	K_d	K_I
Rise Time	Decrease	-	Decrease
Overshoot	Increase	Decrease	Increase
Settling Time	-	Decrease	Increase
Steady-State Error	Decrease	-	Eliminate

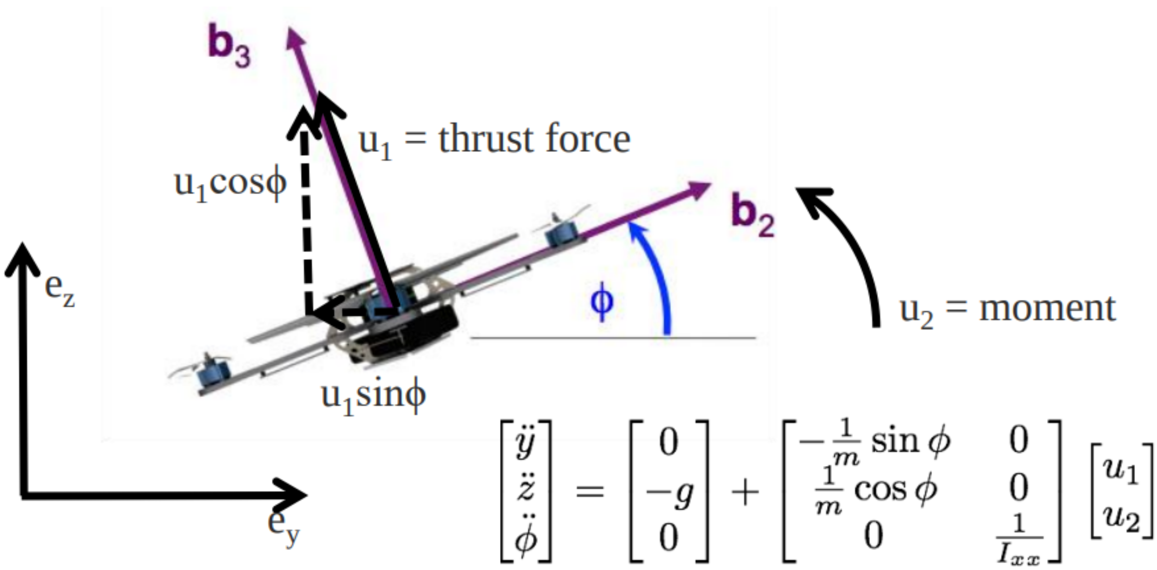
2.5.4 Ziegler-Nichols Method

Heuristic method for PID gain tuning. See this Wikipedia link for more detail.

1. Set $K_d = K_I = 0$.
2. Increase K_p until ultimate gain K_u. K_u is where the system starts to oscillate.
3. Find the oscillation period T_u at K_u.
4. Set gains according to:

Controller	K_p	K_d	K_I
P	$0.5K_u$	--	--
PD	$0.8K_u$	$K_pT_u/8$	--
PID	$0.6K_u$	$K_pT_u/8$	$2K_p/T_u$

2.6 Planar Quadrotor Model



We now derive the dynamics model for a 2-propeller planar quadcopter, that only operates on the y and z axes. If we assume that the left propeller provides a thrust F_1 and right propeller provides a thrust F_2, then let the first input u_1 be the net thrust force so $u_1 = F_1 + F_2$ and let the second input u_2 be the net torque so $u_2 = (F_2 - F_1)\ell$, where ℓ is the length from the center till the propeller. This transformation of the inputs makes it easier for calculating dynamics.

Now using the diagram above, we will derive the dynamics. We note that the only external force is gravity acting in the z direction, which results in the first vector $\begin{bmatrix}0\\ -g\\ 0\end{bmatrix}$. Then, we can see that $u_1 \cos\phi$ is the component of u_1 that affects the z direction and $-u_1\cos\phi$ is the component of u_1 that affects the y direction. Finally, using the physics formula that $\tau = I\alpha$, we get $u_2 = I_{xx}\ddot{\phi}$. Putting it all together and normalizing the force by its mass to get linear acceleration gives us the mentioned dynamics,

$$\begin{bmatrix}\ddot{y}\\ \ddot{z}\\ \ddot{\phi}\end{bmatrix} = \begin{bmatrix}0\\ -g\\ 0\end{bmatrix} + \begin{bmatrix}\frac{-\sin\phi}{m} & 0\\ \frac{\cos\phi}{m} & 0\\ 0 & \frac{1}{I_{xx}}\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix}$$

Letting our state variables be $\vec{x} = \begin{bmatrix} y & z & \phi & \dot{y} & \dot{z} & \dot{\phi} \end{bmatrix}^T$ gives us the overall state space equations:

$$\dot{x} = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -\sin x_3/m & 0 \\ \cos x_3/m & 0 \\ 0 & I_{xx}^{-1} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$