Getting started

This note is written mainly for learning Feedback Control via reading *Feedback Control of Dynamic Systems* and some papers.

Thanks for fellow xzy's support.

Week1

Basic Ideas for Feedback Control

1. Control is the process of making a system variable adhere to a particular value, called the **reference value**. A system designed to follow a changing reference is called **tracking control or a servo**. A system designed to maintain an output fixed regardless of the disturbances present is called a **regulating control or a regulator**.
2. A simple feedback system consists of the **process (or plant)**whose output is to be controlled, the **actuator** whose output causes the process output to change, a reference command signal, and output sensors that measure these signals, and the **controller** that implements the logic by which the control signal that commands the actuator is calculated.
3. A well-designed feedback control system will be **stable, track a desired input or setpoint , reject disturbances, and be insensitive (or robust) to changes in the math model used for design**.
4. The theory and design techniques of control have come to be divided into two categories: **classical control** methods use Laplace transforms (or z-transform) and were the dominant methods for control design until **modern control** methods based on ODEs in state form were introduced into the field starting in the 1960s.

Dynamic models

Step response with Matlab

```
s=tf('s');  % sets up the mode to define the transfer function
sys = (1/1000)/(s + 50/1000);  % defines the transfer function
step(500*sys);   % plots the step response for u = 500.
```

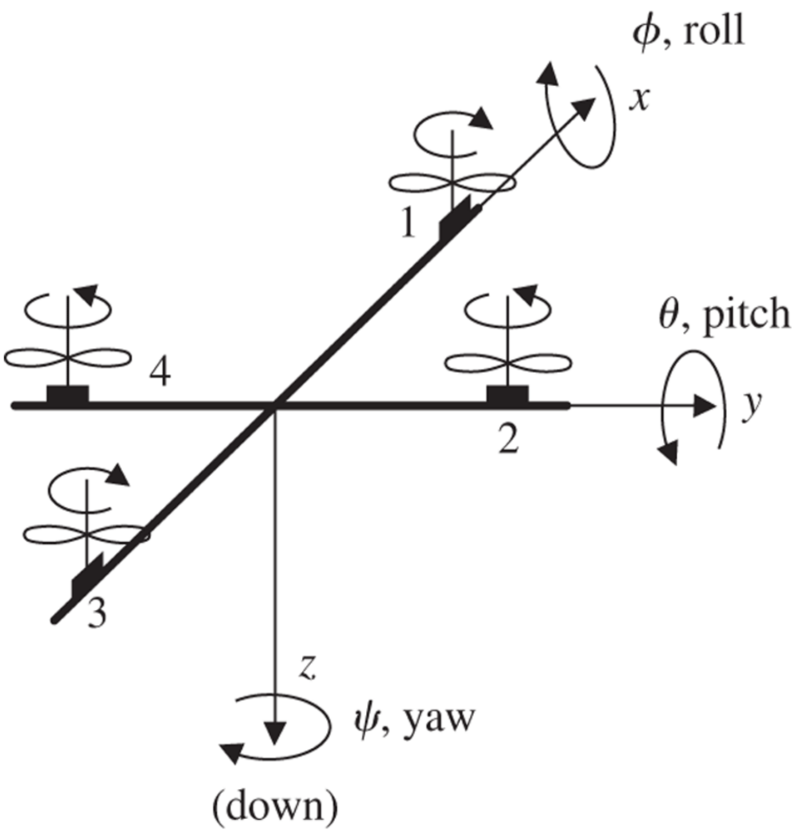
tf() function usage
step() function usage

Collocated control and non-collocated control

The control situation where the sensor and actuator are rigidly attached to one another is called **collocated control**, while the other is called **non-collocated control**

A brief example of Quadrotor Drone

A simple graph of a quadrotor:

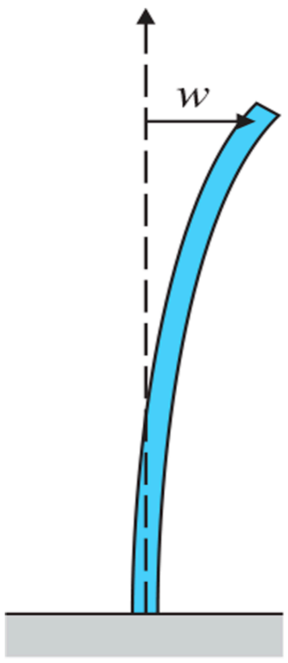


The sensors used in quadrotor are mainly ultrasound(to get the distance from a surface), camera(to detect the motion of the surrounding to judge the motion of drone), IMU, pressure sensor(to get the altitude).
The diagonal rotors(eg. 1 and 3) are driven in the same direction(CW or CCW), the reaction the rotor exerts on the drone consists of the lift force and the reaction torque. To control Pitch and roll, the diagonal rotor must speed up and slow down simultaneously. To control yaw, the diagonal rotor must all speed up or slow down.

The MUX(multiplexer) block in simulink can combine the signal from multi channl and draw them in a same diagram.

Distributed Parameter Systems

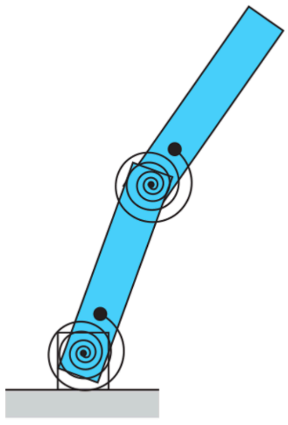
For systems with flexible parts in it(like a rod), the dynamic equation will always a high order equation. eg.



Where

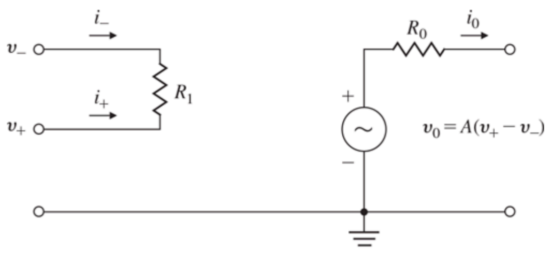
$$\frac{\partial^4 w}{\partial x^4} + \rho \frac{\partial^2 w}{\partial t^2} = 0$$

To simplify the distributed parameter, we can regrad the system as two or more rigid bodies connected by springs, the result is sometimes called a **lumped parameter model**.

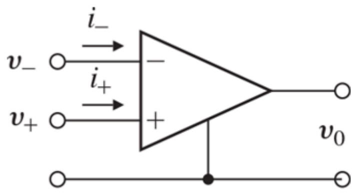


Operational amplifier

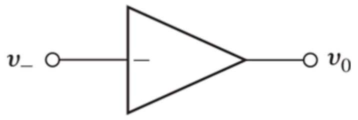
Operational amplifier is basic in circuit system.A operation amplifier's symbol and its schematic are shown below:



(a)



(b)



The third picture means the v+ is connected to the GND.

Basic equation of operational amplifier are:

$$\begin{aligned} i_+ &= i_- = 0 \\ v_+ - v_- &= 0 \end{aligned}$$

Dynamic Reponse

Bode Plots and Matlab implementation

The plot where the frequency response of $\cos(\omega t)$ is drawn with amplitude and phase apart is called bode plot. The following code is an example for bode plot in Matlab

```
k = 1;
tf=('s');
sysH = 1/(s+k);
function
w = logspace(-2,2);
[mag,phase] = bode(sysH,w);
loglog(w,squeeze(mag));
semilogx(w,squeeze(phase));
% define Laplace variable
% define system by its transfer
% set frequency w to 50 values from 10^-2 to 10^2
% compute frequency response
% log–log plot of magnitude
% semi-log plot of phase
```

The usage of several functions :

logspace

bode

squeeze

Some basic functions of Laplace Transform in Matlab

```
%code block 1
num = 2;                        % form numerator
den = poly([-2;-1;-4]);         % form denominator polynomial from its roots
[r,p,k] = residue(num,den);  % compute the residues

%code block 2
s=tf(‘s’); % define Laplace variable
sysH=0.001/(s^2+0.05*s); % form transfer function
p=pole(sysH);      % compute poles
[z,k]=zero(sysH); % compute zeros and transfer function gain

%block 3
s=tf('s'); % define Laplace variable
sysG=0.0002/s ^2; % define system by its transfer function
t=0:0.01:10; % set up time vector with dt = 0.01 sec
% pulse of 25N, at 5 sec, for 0.1 sec duration
u1=[zeros(1,500)
 25*ones(1,10)
zeros(1,491)];     % pulse input
[y1]=lsim(sysG,u1,t); % linear simulation
ff=180/pi; % conversion factor from radians to degrees
y1=ff*y1; % output in degrees
plot(t,u1);     % plot input signal
plot(t,y1);    % plot output response
```

The usage of several functions :

poly

residue

zero   pole

lsim