



**UNSW**  
SYDNEY

# Basic Concepts of Complex System Control

A/Prof Shiyang Tang

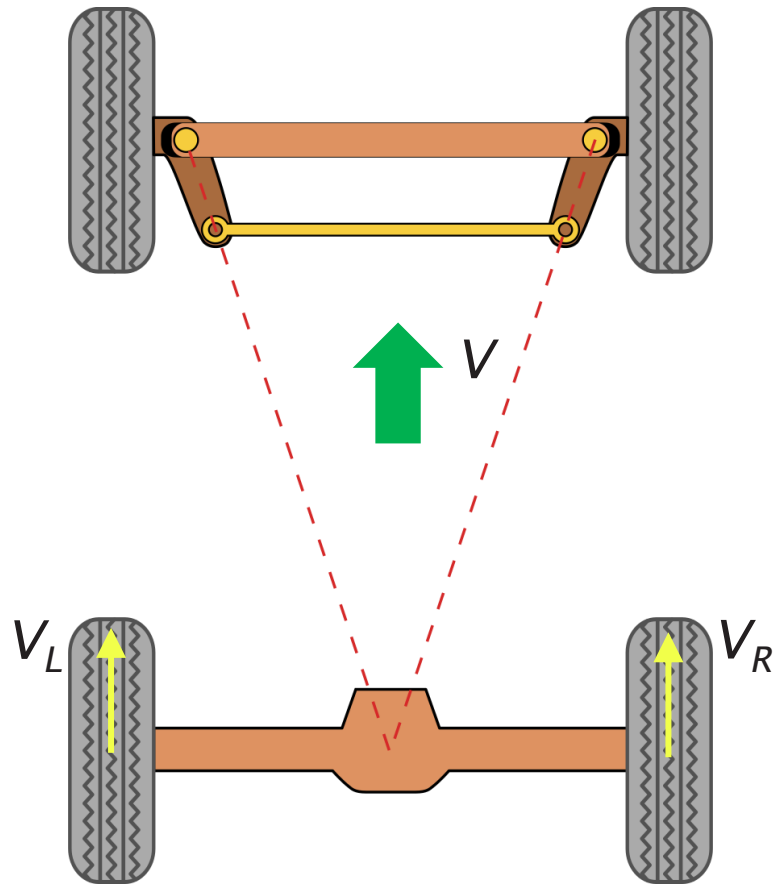
**MTRN3500**

Computing Applications in Mechatronics Systems

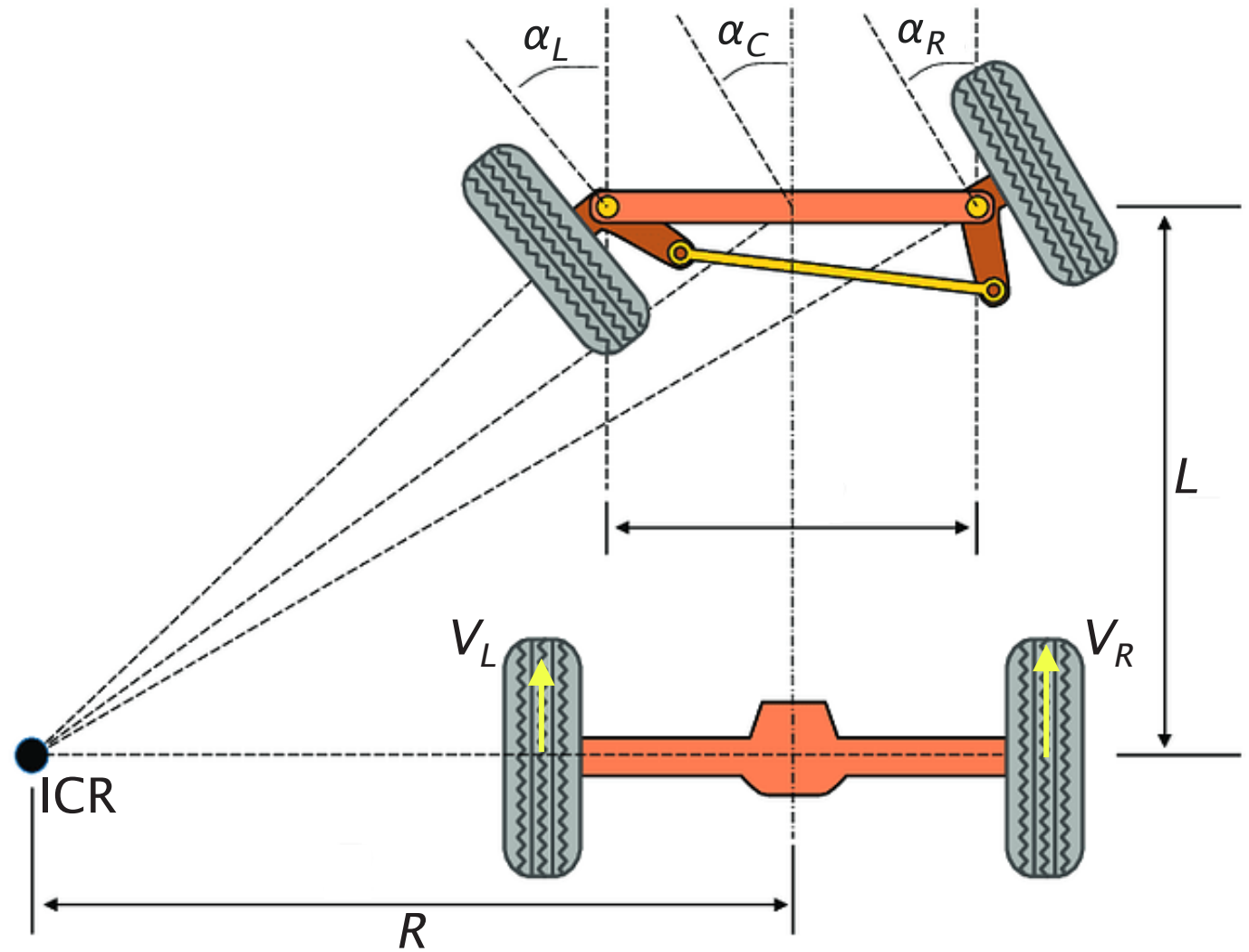
# Rear Wheel Drive



# Ackermann Steering Mechanism



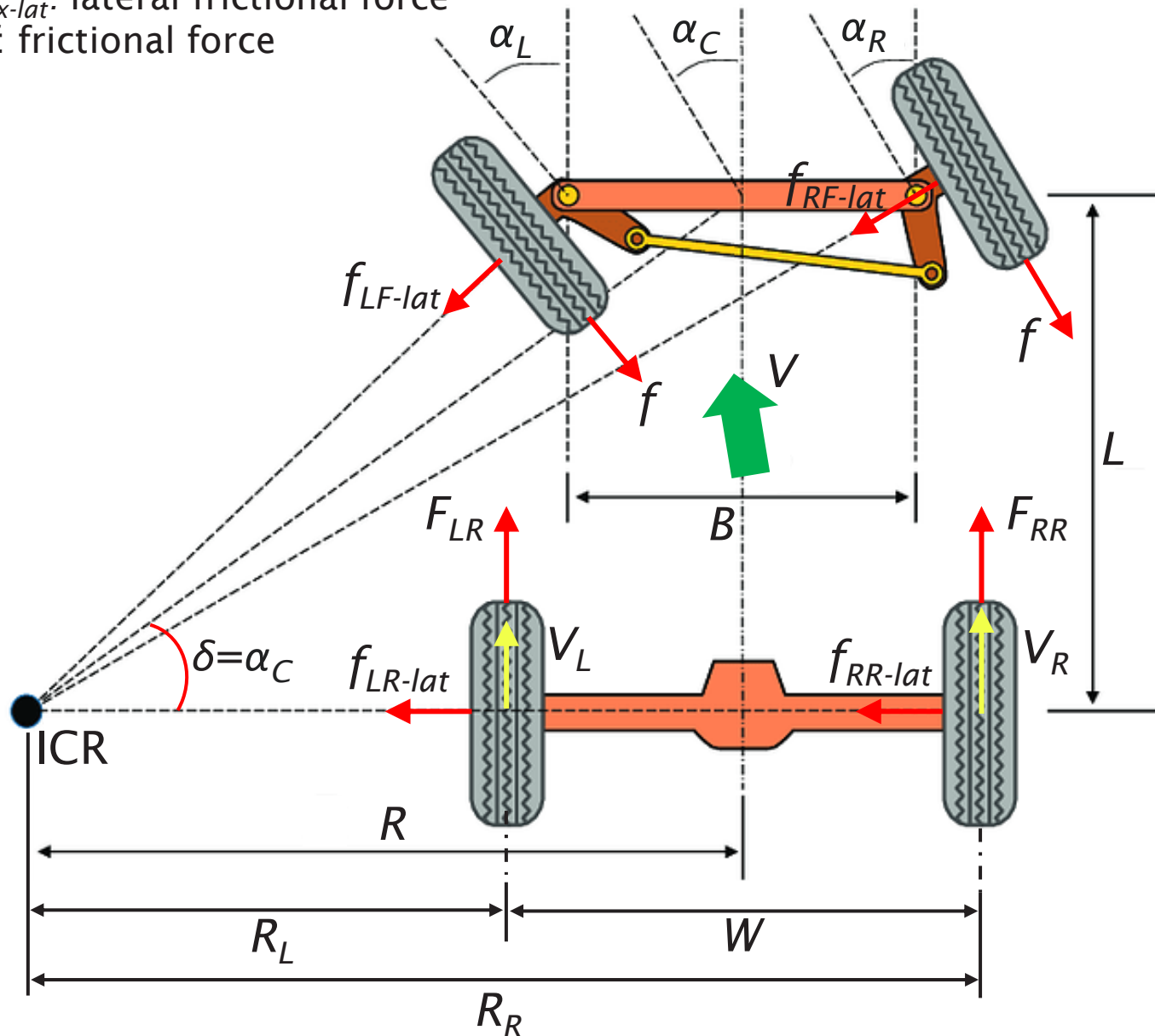
$$V_L = V_R = V$$



$$V_L \neq V_R$$

# Ackermann Steering Mechanism

$F_x$ : Drive force  
 $f_{x-lat}$ : lateral frictional force  
 $f$ : frictional force



$$\tan \delta = \frac{L}{R} \quad V = R\delta'$$

$$\begin{cases} V_L = R_L\delta' = \left(R - \frac{W}{2}\right)\frac{V}{R} = \left(1 - \frac{W}{2R}\right)V \\ V_R = R_R\delta' = \left(R + \frac{W}{2}\right)\frac{V}{R} = \left(1 + \frac{W}{2R}\right)V \end{cases}$$

Replace  $R$

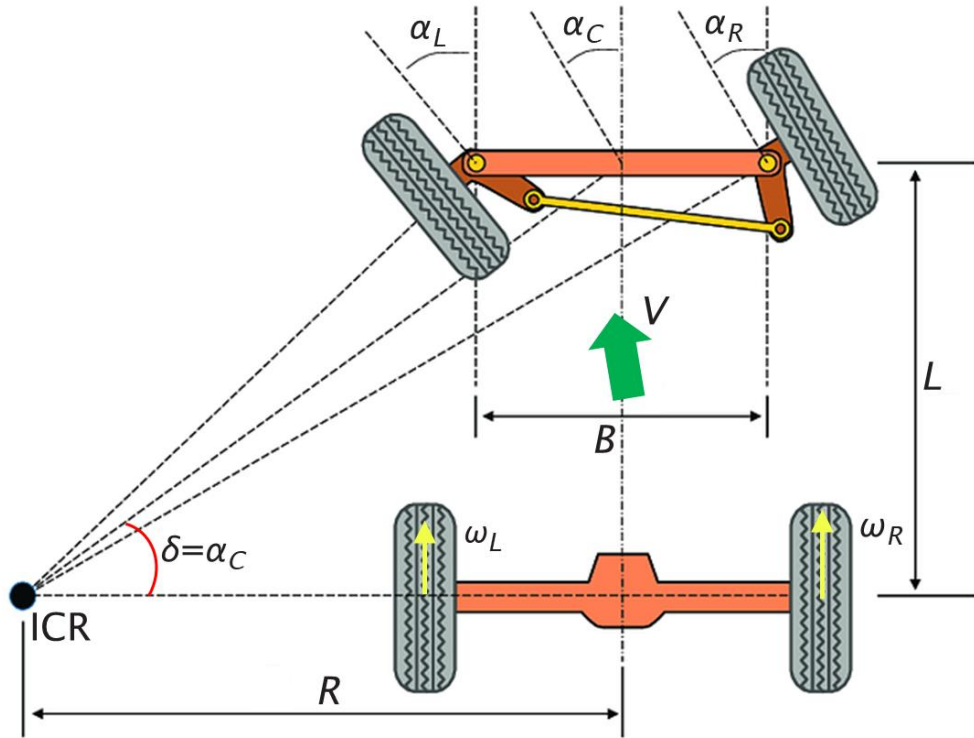
$$\begin{cases} V_L = \left(1 - \frac{W \tan \delta}{2L}\right)V \\ V_R = \left(1 + \frac{W \tan \delta}{2L}\right)V \end{cases}$$

$V_{L/R} = \omega_{L/R} r_{wheel}$

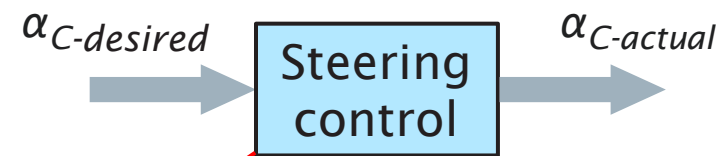
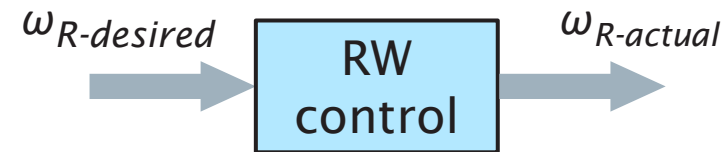
$$\begin{cases} \omega_L = \left(1 - \frac{W \tan \delta}{2L}\right) \frac{V}{r_{wheel}} \\ \omega_R = \left(1 + \frac{W \tan \delta}{2L}\right) \frac{V}{r_{wheel}} \end{cases}$$

# Drive Control Mechanism

Most vehicles do not have independent control of the wheels. Instead, a propeller shaft differential is used to allow the left and right wheels to rotate at different speeds during cornering

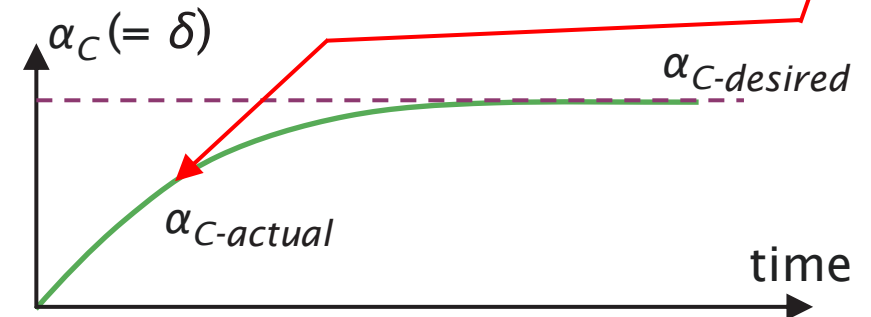
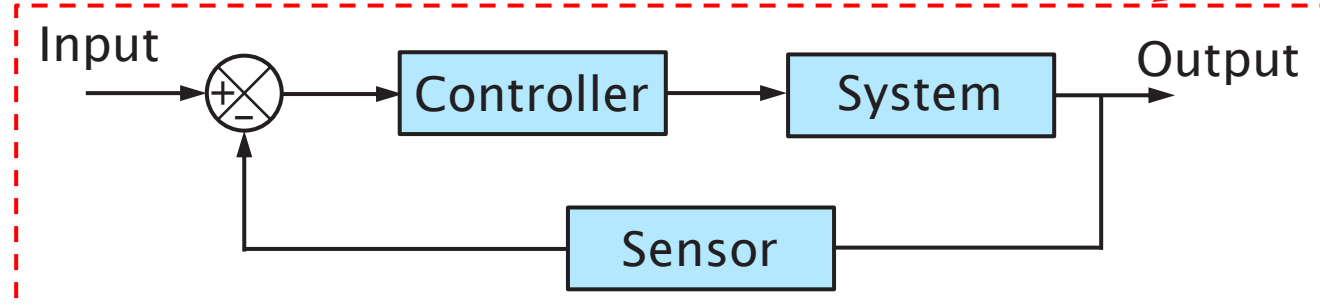


#	<steer>	<speed>	<wdog>	#
	$\alpha_C (= \delta)$	$V$	0/1	



$$\begin{cases} \omega_{L-d} = \left(1 - \frac{W \tan \delta}{2L}\right) \frac{V}{r_{wheel}} \\ \omega_{R-d} = \left(1 + \frac{W \tan \delta}{2L}\right) \frac{V}{r_{wheel}} \end{cases}$$

PID process control loop in Galil



# Hardware Structure

**Embedded computer & Galil controller  
(our attention today, server + client)**

Remote controller

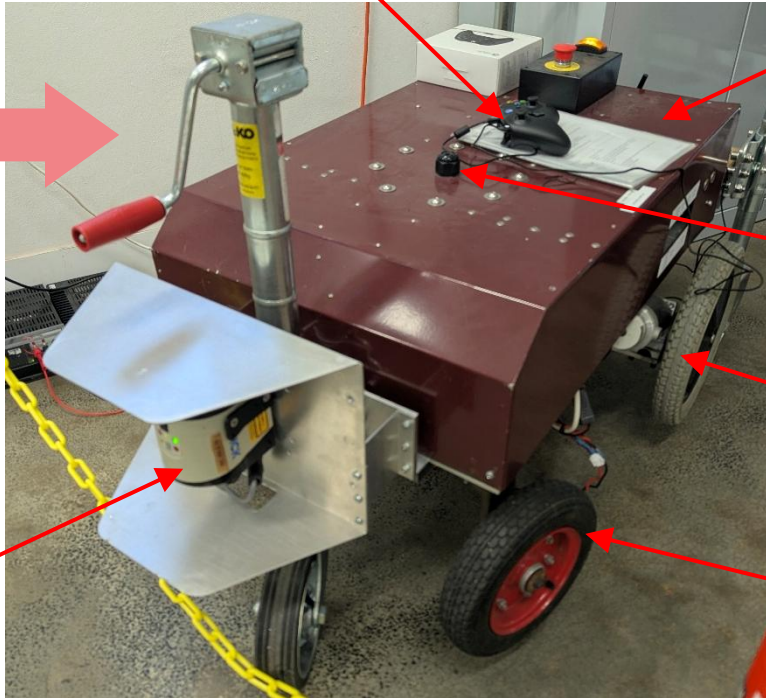
GNSS (removed,  
a server)

Drive wheels

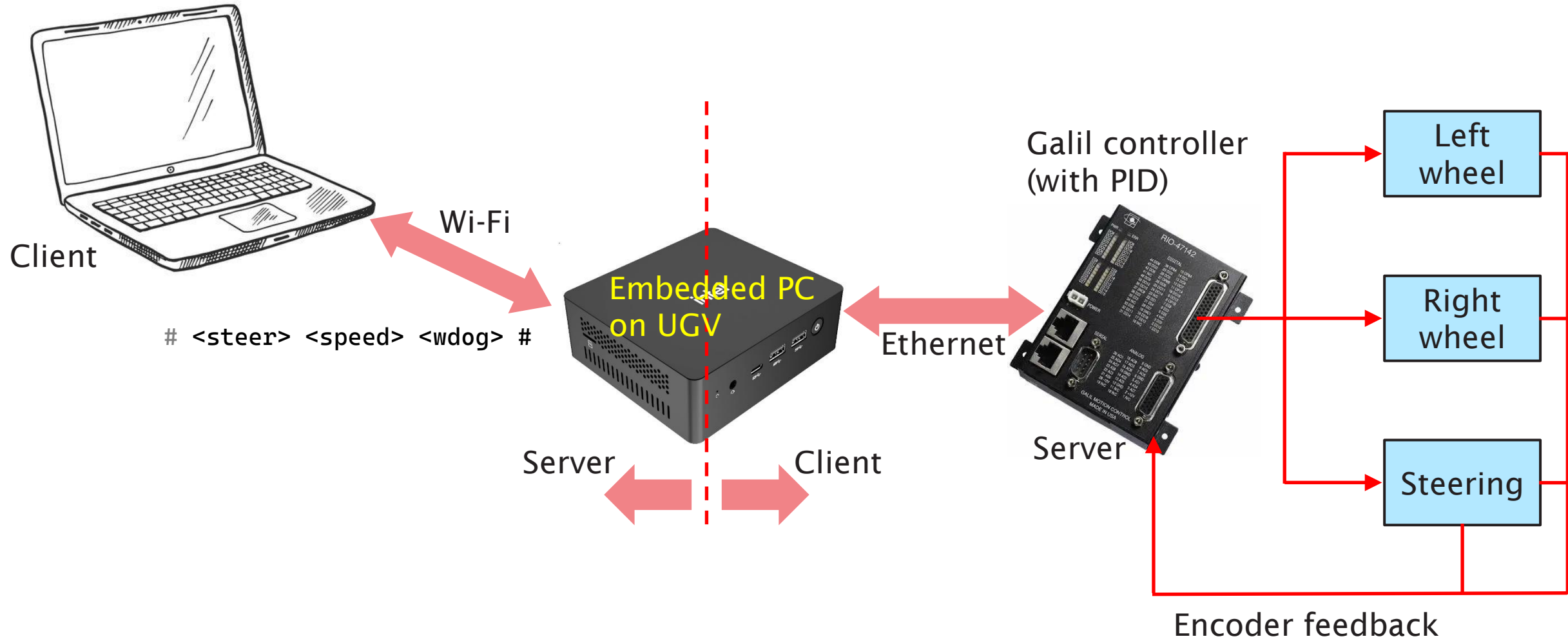
Steered wheels

External  
control PC  
(a client)

LiDAR  
(a server)



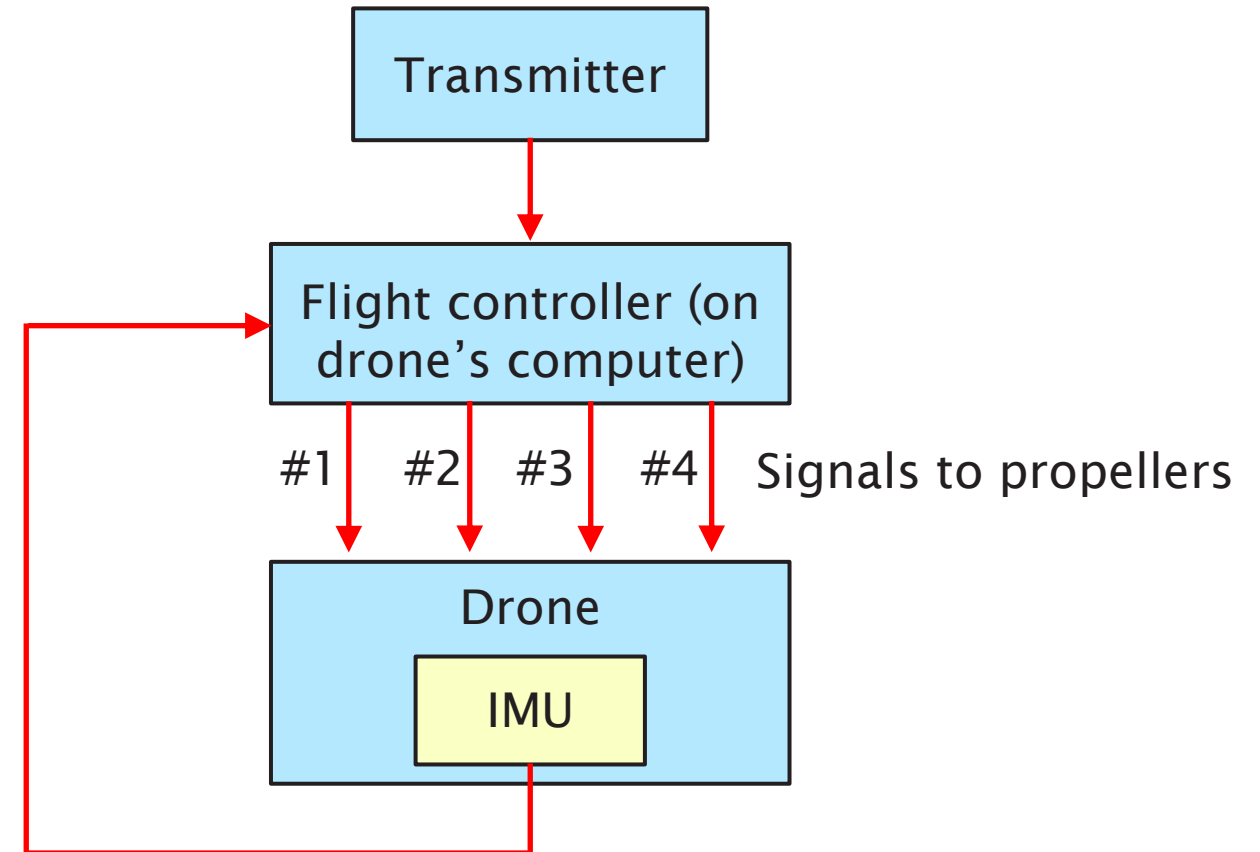
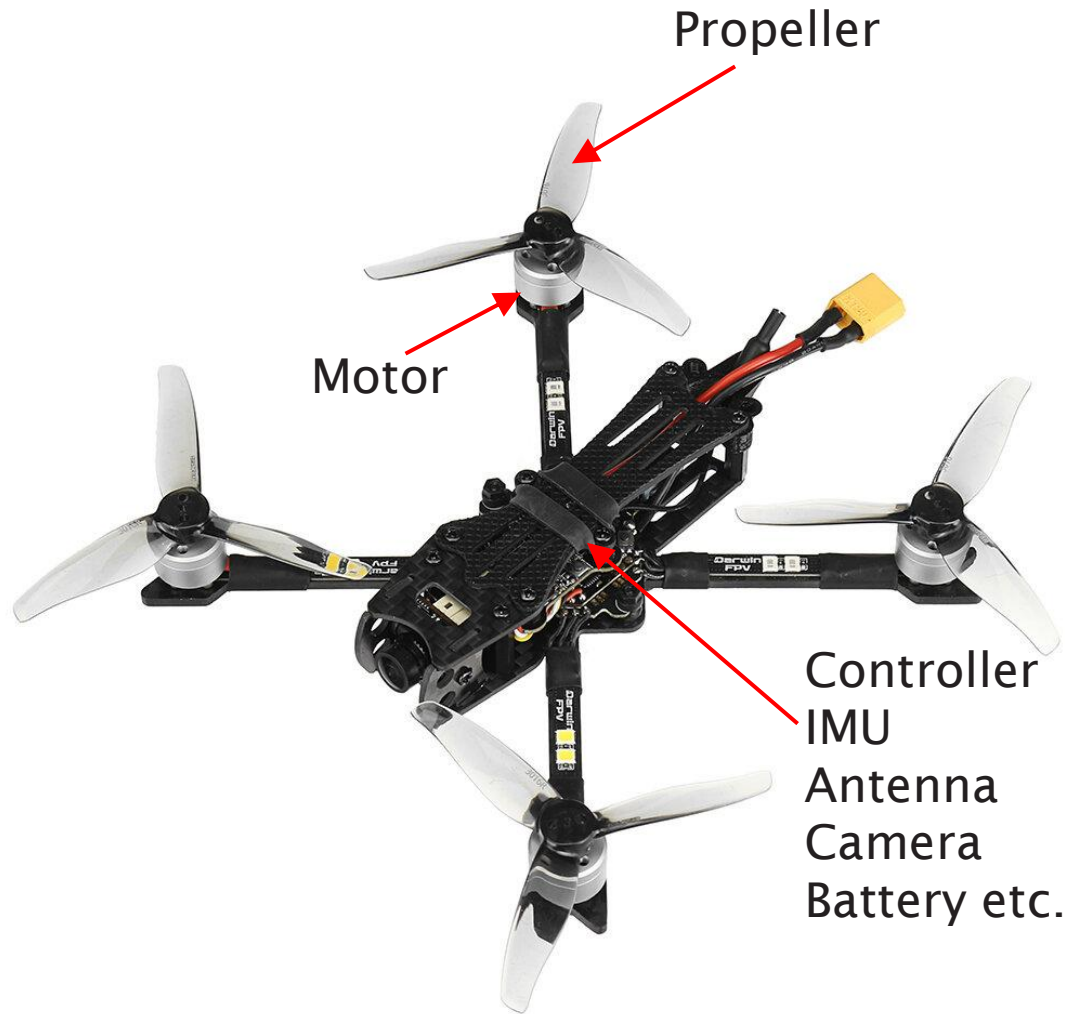
# UGV Driving Network Architecture



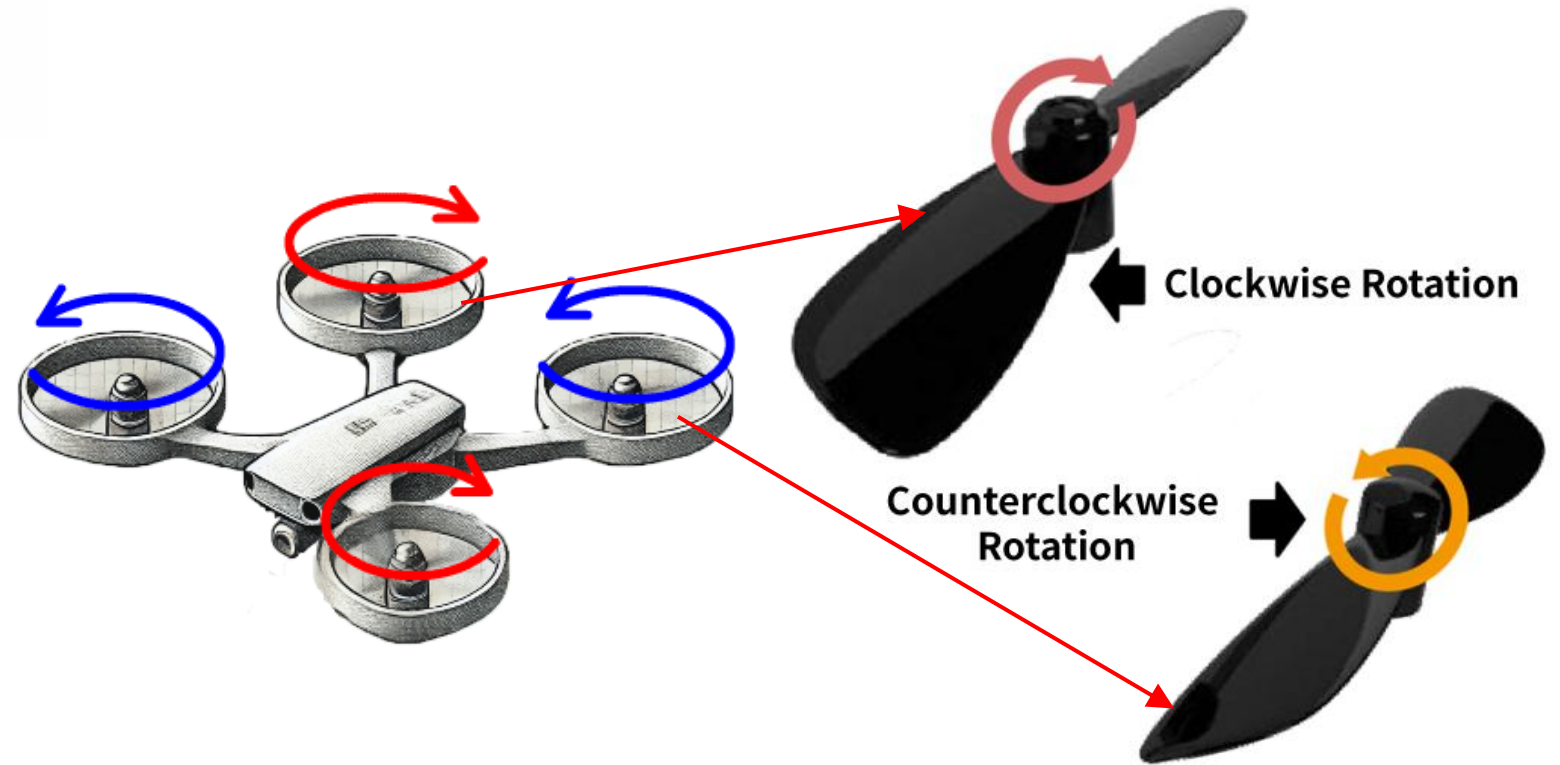
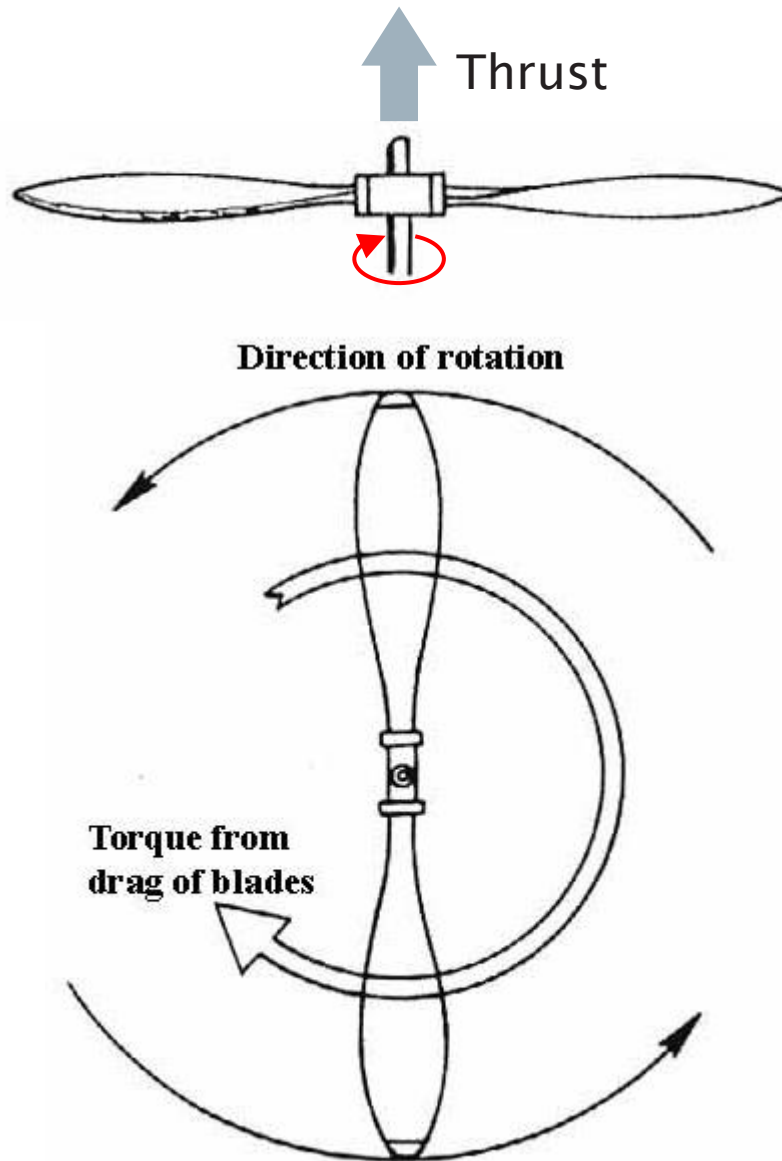
```
int main()
{
    // Instantiate a UGV object (as a client to Galil, or as a server to an external laptop)
    // Connect to Galil
    // Obtain a stream from Galil
    // Start the server
    // While UGV is on
        // Listen to an incoming connection from the laptop client
        // Establish a connection
        // While (connected to external laptop client)
            // Receive data from the client and extract setspeed and set steering
            // Read steering encoder on the Galil to obtain the current steering position
            // Calculate the left wheel speed and the right wheel speed in cts/s (based on the current steering position)
            // Calculate the desired set steering in counts
            // If Watchdog is ok
                // Send commands (desired set steering + calculated wheel speeds) to Galil
            // Else
                // Send zero commands to Galil
            // End - while (connected to external laptop client)
        // Close client connection
    // End - while UGV is on
    // Stop Server
    // Close Galil Client

    return 0;
}
```

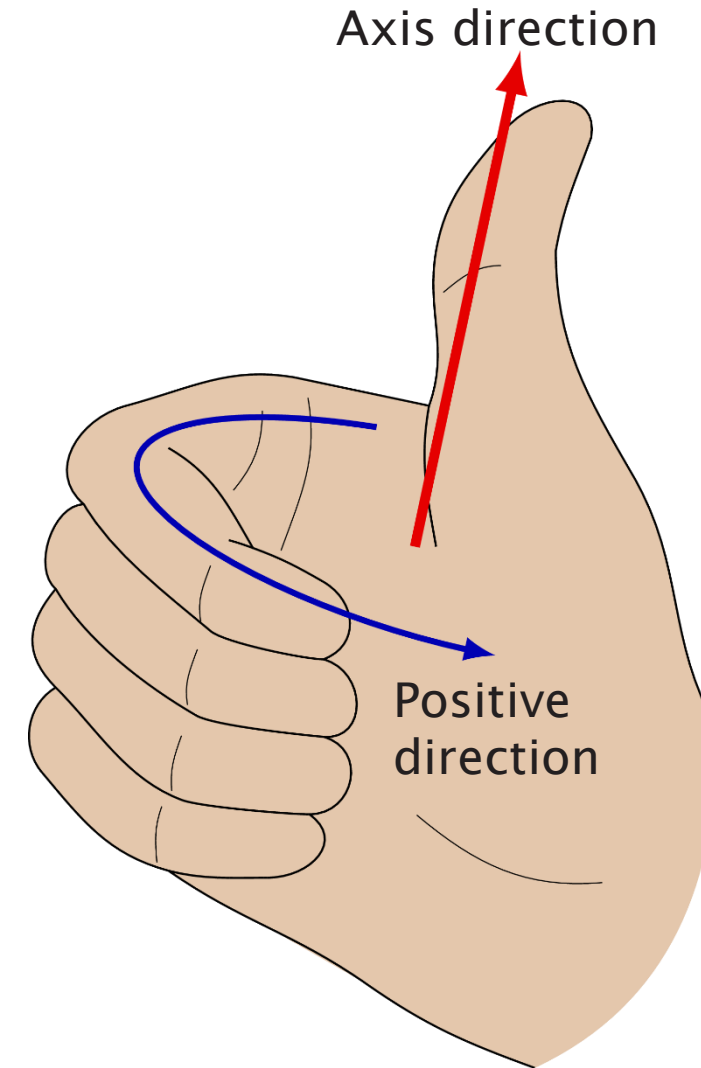
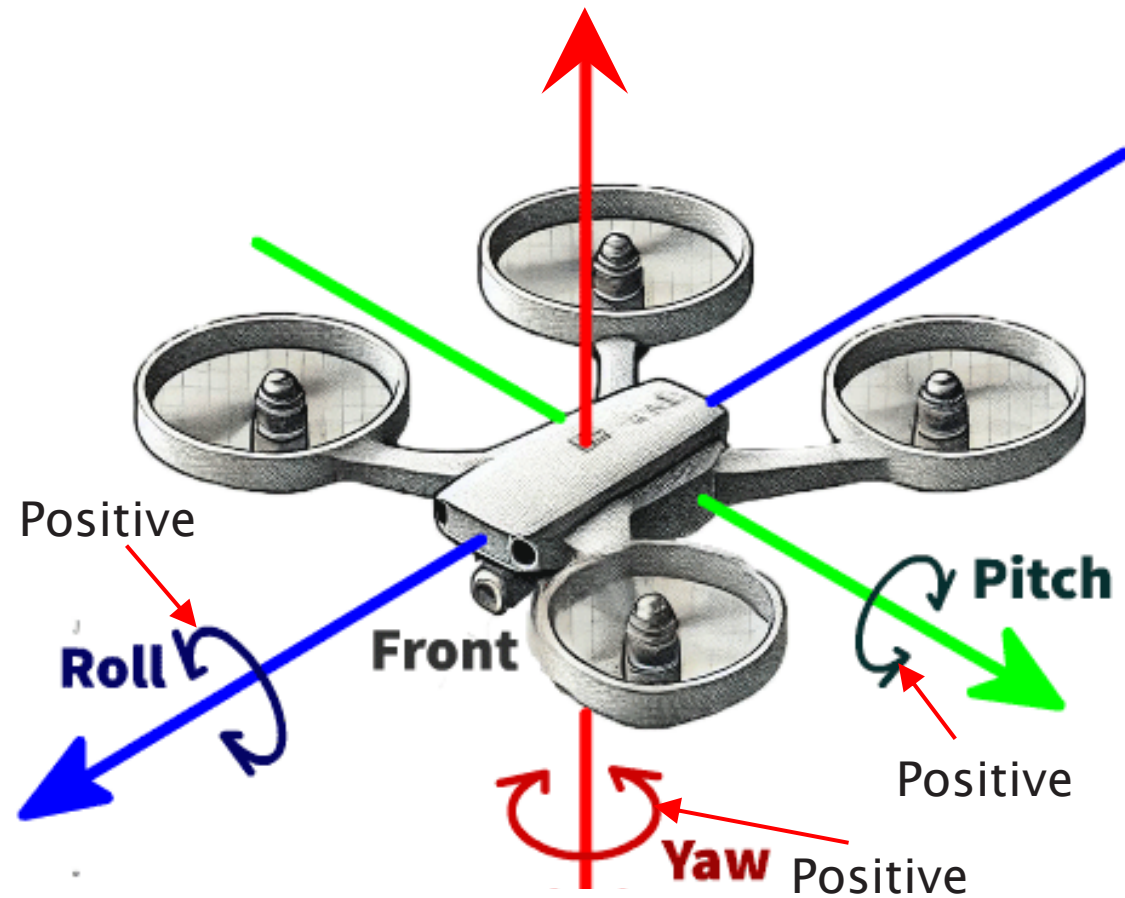
# Quadcopter Control



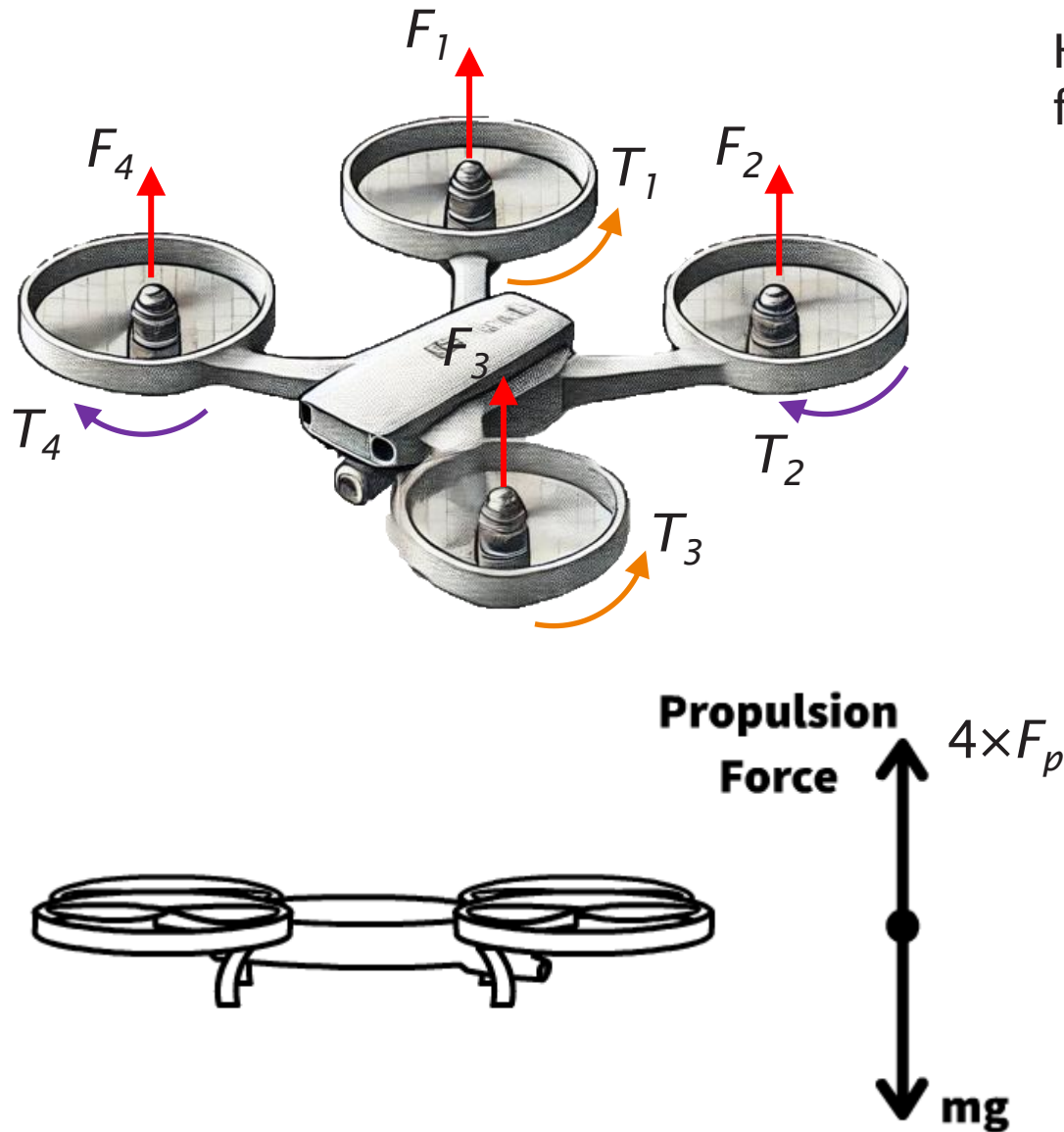
# Quadcopter Torque Effects



# Yaw, Pitch, and Roll



# Height Control



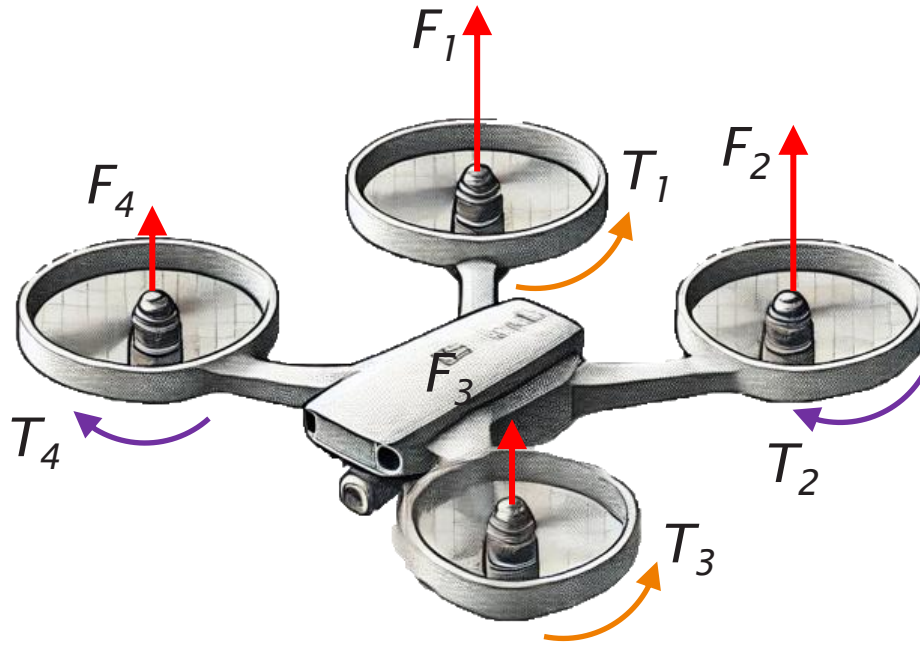
Height control: add an equal amount of force to the four propellers by increasing the rotor speed equally.

$$Force = \begin{cases} F_1 = F_p + \Delta F \\ F_2 = F_p + \Delta F \\ F_3 = F_p + \Delta F \\ F_4 = F_p + \Delta F \end{cases} \Rightarrow \sum_{i=0}^{i=4} F_i = 4(F_p + \Delta F)$$

$$Torque = \begin{cases} T_1 = T_o + \Delta T \\ T_2 = -(T_o + \Delta T) \\ T_3 = T_o + \Delta T \\ T_4 = -(T_o + \Delta T) \end{cases} \Rightarrow \sum_{i=0}^{i=4} T_i = 0$$

$$\begin{cases} F = C_F \omega^2 \\ T = C_T \omega^2 \end{cases}$$

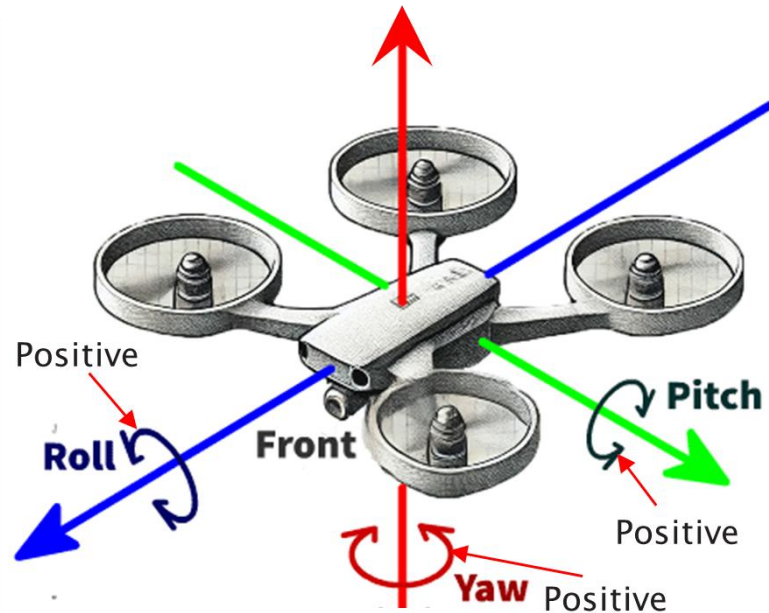
Propeller angular velocity



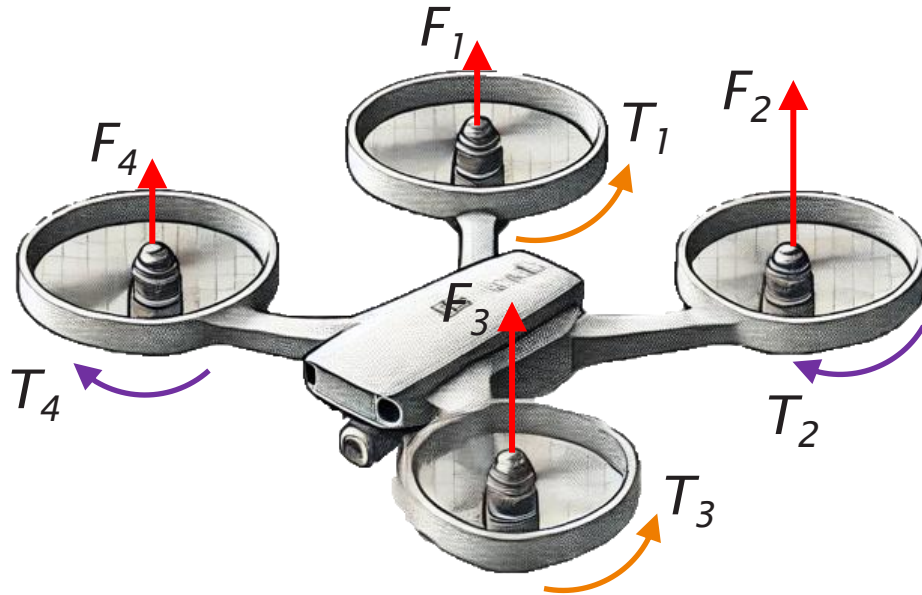
$$Force = \begin{cases} F_1 = \frac{F_p + \Delta F}{\cos \theta} \\ F_2 = \frac{F_p + \Delta F}{\cos \theta} \\ F_3 = \frac{F_p - \Delta F}{\cos \theta} \\ F_4 = \frac{F_p - \Delta F}{\cos \theta} \end{cases} \Rightarrow \sum_{i=1}^4 F_i = \frac{4F_p}{\cos \theta}$$

Gravity compensation

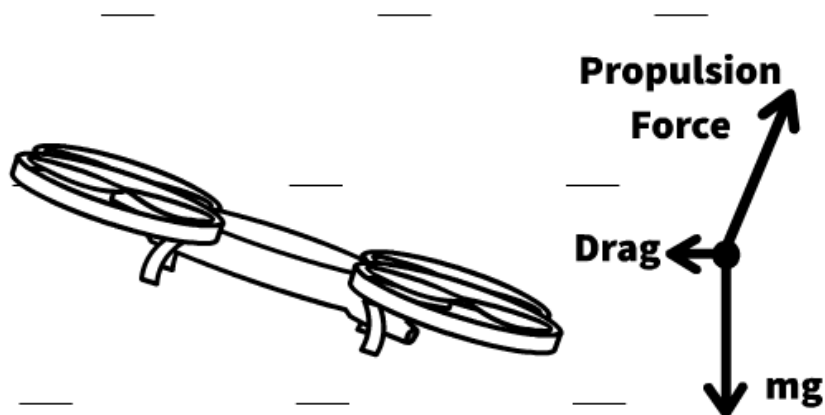
Pitch angle



$$Torque = \begin{cases} T_1 = T_o + \Delta T \\ T_2 = -(T_o + \Delta T) \\ T_3 = T_o - \Delta T \\ T_4 = -(T_o - \Delta T) \end{cases} \Rightarrow \sum_{i=1}^4 T_i = 0$$

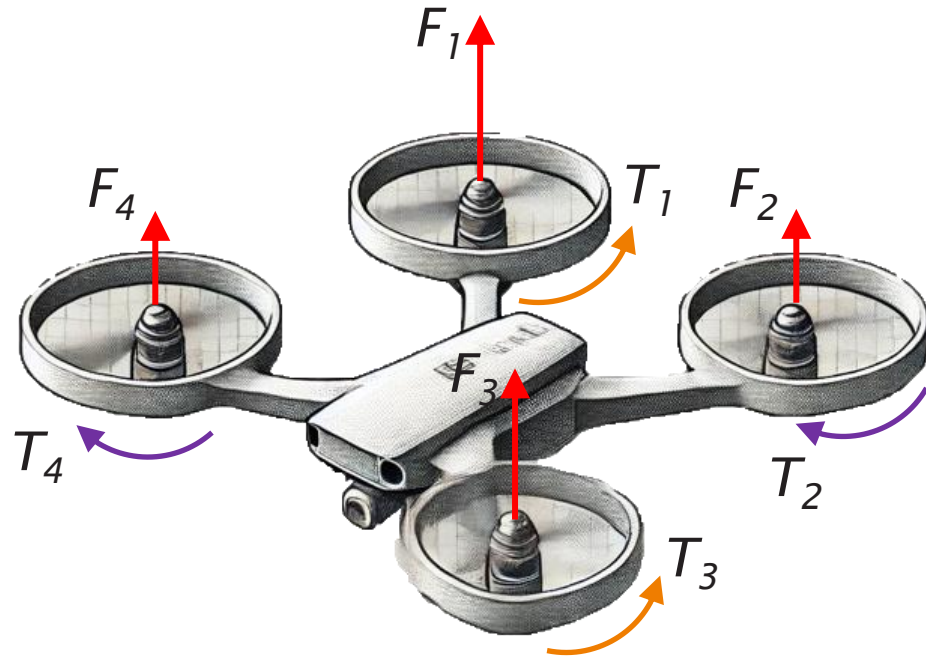


$$Force = \begin{cases} F_1 = \frac{F_p - \Delta F}{\cos \phi} \\ F_2 = \frac{F_p + \Delta F}{\cos \phi} \\ F_3 = \frac{F_p + \Delta F}{\cos \phi} \\ F_4 = \frac{F_p - \Delta F}{\cos \phi} \end{cases} \Rightarrow \sum_{i=0}^{i=4} F_i = \frac{4F_p}{\cos \phi}$$



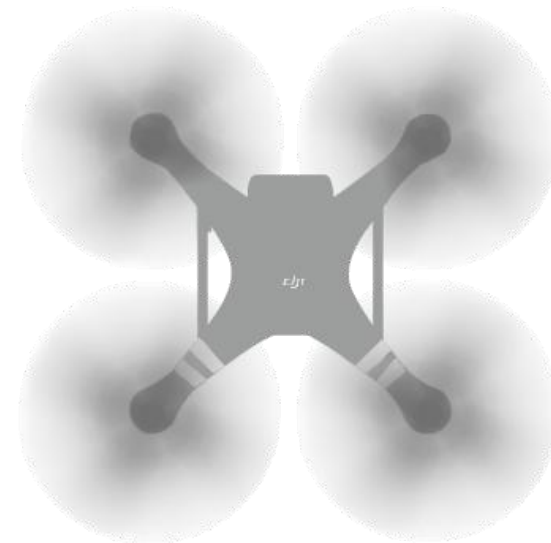
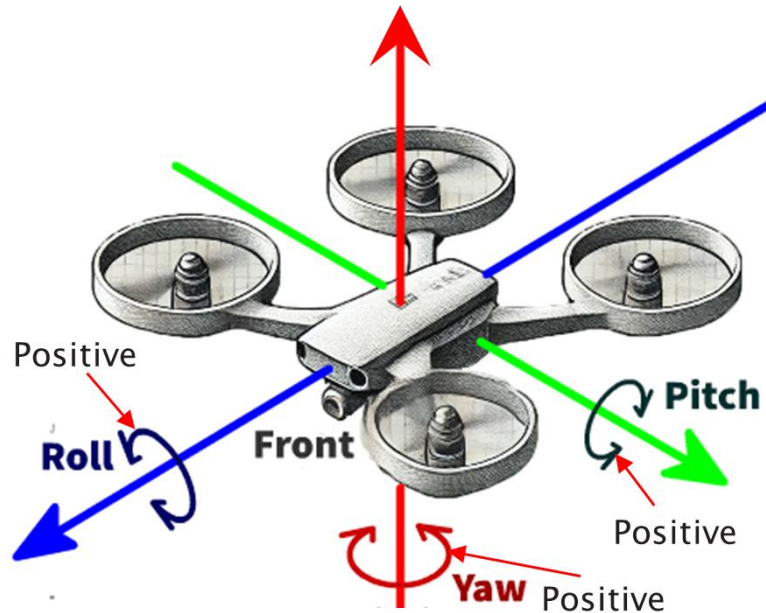
$$Torque = \begin{cases} T_1 = T_o - \Delta T \\ T_2 = -(T_o + \Delta T) \\ T_3 = T_o + \Delta T \\ T_4 = -(T_o - \Delta T) \end{cases} \Rightarrow \sum_{i=0}^{i=4} T_i = 0$$

# Yaw Control



$$\text{Force} = \begin{cases} F_1 = F_p + \Delta F \\ F_2 = F_p - \Delta F \\ F_3 = F_p + \Delta F \\ F_4 = F_p - \Delta F \end{cases} \Rightarrow \sum_{i=0}^{i=4} F_i = 4F_p = mg$$

$$\text{Torque} = \begin{cases} T_1 = T_o + \Delta T \\ T_2 = -(T_o - \Delta T) \\ T_3 = T_o + \Delta T \\ T_4 = -(T_o - \Delta T) \end{cases} \Rightarrow \sum_{i=0}^{i=4} T_i = 4\Delta T$$



# Control Diagram

