Lift Models

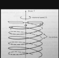
Two fundamental actuation/lift mechanisms: wing and rotor

Quaternions

Rotor Lift Model:

Rotor rotation generates both thrust and torque

$$T = K_1 \rho n^2 D^4$$
$$\tau = K_2 \rho n^2 D^5$$



T: thrust
τ: torque
ρ: air density (kg/m3)
n: rotation speed (rev/s)
D: diameter (m)

In practice, we use the equations

$$T \propto n^2$$
$$\tau = k_{T2\tau} T$$

$k_{T2\tau}$ is called thrust to torque ratio

It is typically very small. This is why controlling the yaw is slow

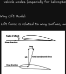
Why is torque produced?
As the rotor blades spin through the air, they not only generate lift but also experience drag. Drag is a resistive force that opposes the motion of the blades, and this force acts tangentially to the rotor disk. The drag creates a rotational resistance, resulting in torque that tries to slow down the rotor's spinning motion.

- The spinning blades create a low-pressure area above the propeller as the air is pulled downwards.
- The air is accelerated and pushed down, creating a high-pressure area below the propeller.
- This difference in pressure (low pressure above and high pressure below the propeller) generates an upward force, which provides lift to the drone.

High-levels rules of thumb when designing multi-rotors:
- Large propellers are more efficient
- Slower rpm is more efficient (but less agile)
- constant rpm becomes important as resonances/stiffness in the frame aligns with vehicle modes (especially for helicopters)

Wing Lift Model:

Lift force is related to wing surface, angle of attack, air density, airspeed, etc.



The shape of the wing (airfoil) is designed so that air moving over the top surface travels faster than the air moving under the bottom surface. The top of the wing is usually curved, and the bottom is relatively flat.

According to Bernoulli's principle, as the speed of airflow increases, the pressure decreases. Since the air moves faster over the top surface, the pressure is lower above the wing compared to below it, where the airflow is slower.

This pressure difference (lower pressure above the wing, higher pressure below the wing) creates an upward force known as lift, which helps keep the aircraft in the air.

2D Planar Drone Modeling

p: position of CoM in world frame
v: velocity of CoM in world frame
θ: angle
ω: angular rate
T_1, T_2: thrust from each rotor
$\vec{g} = [0; -g]$ gravity vector
m, J, d: mass, inertia, arm length

Translational kinematics:
$$\dot p = v$$
Translational dynamics:
$$m\dot v = m\vec g + (T_1+T_2)\begin{bmatrix}-\sin\theta\\ \cos\theta\end{bmatrix}$$
Rotational kinematics:
$$\dot\theta = \omega$$
Rotational dynamics:
$$J\dot\omega = (T_2 - T_1)d$$

Total thrust $T = T_1 + T_2$

Torque $\tau = (T_1 - T_2)\cdot d$

Actuation matrix $\begin{bmatrix}T\\ \tau\end{bmatrix} = \begin{bmatrix}1 & 1\\ -d & d\end{bmatrix}\begin{bmatrix}T_1\\ T_2\end{bmatrix}$

Full rank. Actuation matrix for quadrotor is 4x4

Control and state
$$x = \begin{bmatrix}p\\ v\\ \theta\\ \omega\end{bmatrix} \quad u = \begin{bmatrix}T\\ \tau\end{bmatrix}$$
$$x \in \mathbb{R}^6 \quad u \in \mathbb{R}^2$$

In body frame:

Total force in body frame: $\begin{bmatrix}0\\ T\end{bmatrix}$

$$R = \begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}$$

Total force in world frame: $R\begin{bmatrix}0\\ T\end{bmatrix} = Re_2T \quad e_2 = \begin{bmatrix}0\\ 1\end{bmatrix}$

Properties:

- 3 DOF: x, y, θ
- Has 2 actuators: thrust force, torque for rotation
- Underactuated system: mismatch between the number of degrees of freedom (3) and the number of actuators (2). System cannot independently control all its states.

In hovering condition,

$\theta = 0$ no rotation
$\omega = 0$ no angular velocity
$v = 0$ no linear velocity

This represents the drone maintaining a stable position in air without any movement

From this hovering state, the drone cannot directly generate acceleration in the horizontal direction. This is because:
1. The main thrust force is aligned vertically with the drone's body when θ = 0
2. There's no direct actuator to push the drone sideways.

To move horizontally, the drone must execute a two-step process:
1. Turn: The drone needs to rotate (change θ) to tilt its body.
2. Accelerate: Once tilted, the vertical thrust will have a horizontal component, allowing for horizontal acceleration.

This process is fundamentally different from a double integrator system, where you can directly apply force in any direction.

The horizontal motion is coupled with the rotational dynamics. You can't model horizontal movement independently of rotation.

Quadrotor Modeling

{A} is the right-hand world frame (inertia frame)
{B} is the right-hand body frame with unit vectors b_1, b_2, b_3
b_1, b_2, b_3 are the axes of {B} w.r.t {A}

Euler angle parameterization (Z-X-Y):
Roll: φ
Pitch: θ
Yaw: ψ

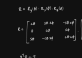
$$R_z(\psi) = \begin{bmatrix}c\psi & s\psi & 0\\ -s\psi & c\psi & 0\\ 0 & 0 & 1\end{bmatrix} \quad R_y(\theta) = \begin{bmatrix}c\theta & 0 & -s\theta\\ 0 & 1 & 0\\ s\theta & 0 & c\theta\end{bmatrix} \quad R_x(\phi) = \begin{bmatrix}1 & 0 & 0\\ 0 & c\phi & s\phi\\ 0 & -s\phi & c\phi\end{bmatrix}$$

$$R = R_y(\theta)\cdot R_x(\phi)\cdot R_z(\psi)$$

$$R = \begin{bmatrix}c\theta & s\theta s\phi & -s\theta c\phi\\ 0 & c\phi & s\phi\\ s\theta & -c\theta s\phi & c\theta c\phi\end{bmatrix}\begin{bmatrix}c\psi & s\psi & 0\\ -s\psi & c\psi & 0\\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}c\psi c\theta - s\phi s\psi s\theta & c\theta s\psi + c\psi s\phi s\theta & -c\phi s\theta\\ -c\phi s\psi & c\phi c\psi & s\phi\\ c\psi s\theta + c\theta s\phi s\psi & s\psi s\theta - c\psi c\theta s\phi & c\phi c\theta\end{bmatrix}$$

$$R^T R = I$$

$Re_1 = b_1$
$Re_2 = b_2$
$Re_3 = b_3$



Torque from each motor: $\tau_i = k_{T2\tau} T_i$

$$\begin{bmatrix}T\\ \tau_1\\ \tau_2\\ \tau_3\end{bmatrix} = \begin{bmatrix}1 & 1 & 1 & 1\\ 0 & d & 0 & -d\\ -d & 0 & d & 0\\ k & -k & k & -k\end{bmatrix}\begin{bmatrix}T_1\\ T_2\\ T_3\\ T_4\end{bmatrix}$$

p ∈ {A}: position of CoM in world frame.
g = [0; 0; -g]
v ∈ {A}: linear velocity of {B} w.r.t frame {A} expressed in {A}
ω ∈ {B}: angular velocity of {B} w.r.t frame {A} expressed in {B}
J: inertia matrix expressed in {B}
Total force T and total torque τ ∈ {B}

Angular velocity can be represented using the skew-symmetric matrix S(ω), which encodes the cross-product operation.

$$S(\omega) = \begin{bmatrix}0 & -\omega_3 & \omega_2\\ \omega_3 & 0 & -\omega_1\\ -\omega_2 & \omega_1 & 0\end{bmatrix}$$

$$S(\omega)v = \omega \times v$$

Consider a vector v_b in the body frame. The corresponding vector in the world frame is

$$v = Rv_b$$
$$\frac{d}{dt}(v) = \frac{d}{dt}(Rv_b) = \dot R v_b$$
$$\dot R v_b = \omega \times (Rv_b) = S(\omega)Rv_b$$
$$\dot R = S(\omega)R$$
$$\dot R = RS(\omega_b)$$

Angular momentum $L = J\omega$

$$\frac{d}{dt}(L) = J\dot\omega + \omega\times J\omega$$

$L = J\omega$ in body frame

When an object is rotating, it experiences additional effects due to its own angular velocity. These are often described using the gyroscopic effect. Jω × ω represents the gyroscopic torque or the torque generated by the angular momentum of the rotating body itself.

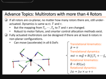
$$J\dot\omega = J\omega\times\omega + \tau$$

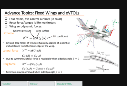
Translational kinematics:
$$\dot p = v$$
Translational dynamics:
$$m\dot v = m\vec g + Re_3T$$
Rotational kinematics:
$$\dot R = RS(\omega)$$
Rotational dynamics:
$$J\dot\omega = J\omega\times\omega + \tau$$

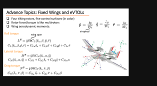
Control and state
$$x = \begin{bmatrix}p\\ v\\ R\\ \omega\end{bmatrix} \quad u = \begin{bmatrix}T\\ \tau\end{bmatrix}$$
$$x \in \mathbb{R}^{18} \quad u \in \mathbb{R}^4$$ 4 underactuated

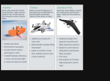
Quaternion q is used to represent rotations.
$$q = [q_w, q_x, q_y, q_z]$$

Advanced Topics









2D Drone: Linearization

In hovering condition,

$\theta = 0$ $T = g$
$\omega = 0$ $\tau = 0$
$v = 0$

Translational kinematics:
$$\dot p = v$$
Translational dynamics:
$$m\dot v = m\vec g + (T_1+T_2)\begin{bmatrix}-\sin\theta\\ \cos\theta\end{bmatrix}$$
Rotational kinematics:
$$\dot\theta = \omega$$
Rotational dynamics:
$$J\dot\omega = (T_2 - T_1)d$$

Control and state
$$x = \begin{bmatrix}p\\ v\\ \theta\\ \omega\end{bmatrix} \quad u = \begin{bmatrix}T\\ \tau\end{bmatrix}$$

$$\dot x = \begin{bmatrix}\dot p\\ \dot v\\ \dot\theta\\ \dot\omega\end{bmatrix} = \underbrace{\begin{bmatrix}v\\ \vec g + T\begin{bmatrix}-\sin\theta\\ \cos\theta\end{bmatrix}\\ \omega\\ \tau\end{bmatrix}}_{f}$$

$$\begin{array}{c|c} \frac{\partial f}{\partial x} = \begin{bmatrix}0 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 0 & -T\cos\theta & 0\\ 0 & 0 & 0 & 0 & -T\sin\theta & 0\\ 0 & 0 & 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix} & \begin{matrix}p_1\\ p_2\\ v_1\\ v_2\\ \theta\\ \omega\end{matrix}\\ \begin{matrix}p_1 & p_2 & v_1 & v_2 & \theta & \omega\end{matrix} & \end{array} \quad \frac{\partial f}{\partial u} = \begin{bmatrix}0 & 0\\ 0 & 0\\ -\sin\theta & 0\\ \cos\theta & 0\\ 0 & 0\\ 0 & 1\end{bmatrix}$$

2D Drone: Cascaded Control

Goal: To track a desired position setpoint p_d using a two-stage control approach.

Step 1: Position controller (Outer Loop)

The position control loop generates a force f_d to track the desired position p_d.
The dynamics of the 2D drone position are:

$$\dot p = v$$
$$\dot v = \vec g + f_d$$
$$f_d = -\vec g - K_p(p - p_d) - K_D v$$

This is a PD controller to compute the desired force f_d.
This control law produces a force f_d that compensates for gravity and corrects position errors, driving the 2D drone toward the desired position p_d.

Step 2: Convert fd into desired thrust T and desired attitude θ_d

$$\theta_d = -\mathrm{atan}\left(\frac{f_{dx}}{f_{dy}}\right)$$

The total thrust T is the projection of the desired force fd onto the body frame's z-axis

$$T = f_d^T\begin{bmatrix}-\sin\theta\\ \cos\theta\end{bmatrix}$$

Step 3: Attitude Control (Inner Loop)

The attitude control loop is responsible for tracking the desired attitude θ_d determined above.

$$\tau = -K_{p\tau}(\theta - \theta_d) - K_{D\tau}\omega$$

This PD controller adjusts the 2D drone's angular velocity and orientation to track the desired attitude.

Step 4: Map [T; τ] to [T_1; T_2]

Total thrust $T = T_1 + T_2$

Torque $\tau = (T_1 - T_2)\cdot d$

Frequency: outer loop position controller is running much slower (50-100Hz) than the inner loop attitude controller (500Hz), so we can assume $\theta \to \theta_d$ very fast!

2D Drone: Trajectory Tracking

For continuously tracking a trajectory $p_d(t)$, we update the feedforward term

$$f_d = -\vec g - K_p(p - p_d) - K_D(v - v_d) + a_d \qquad v_d = \dot p_d$$
$$\theta_d = -\mathrm{atan}\left(\frac{f_{dx}}{f_{dy}}\right) \quad T = f_d^T\begin{bmatrix}-\sin\theta\\ \cos\theta\end{bmatrix} \qquad a_d = \ddot p_d$$
$$\tau = -K_{p\tau}(\theta - \theta_d) - K_{D\tau}(\omega - \omega_d) + \tau_d$$

ω_d will depend on jerk ($\dddot p_d$) and depend on snap ($\ddddot p_d$).
Can be ignored if the jerk or snap is big

Quadrotor: Cascaded Control

Direct linearization is a bad idea

Goal: continuously tracking a trajectory or setpoint $p_d(t)$

Step 1: Position Control

$$f_d = -\vec g - K_p(p - p_d) - K_D(v - v_d) + a_d$$

Step 2: Desired Attitude

Then convert fd to T

$$Re_3 = z$$

T is the projection of fd on the z-axis: $T = f_d^T z$

This desired force fd defines the direction in which the quadrotor should orient itself (along the desired z-axis).

$$z_d = \frac{f_d}{\|f_d\|}$$

This is the direction of the force needed to compensate for gravity and control errors.

Desired yaw angle: The yaw angle ψ_d is typically an additional degree of freedom that can be controlled. In most cases, it is set to 0 or maintained constant to ensure a stable heading.

The desired rotation matrix R_d combines the desired orientation along z-axis and the desired yaw angle.