



Drone modeling

Aerodynamics, Dynamics, Control



Last update: December 21, 2023

Agenda

- Basic principles of aerodynamics
- How propellers work
- Drone design and flight principles
- Dynamic modeling
- Control



Quadrotor drones

What is a quadrotor?



Quadrotor drones

What is a quadrotor?

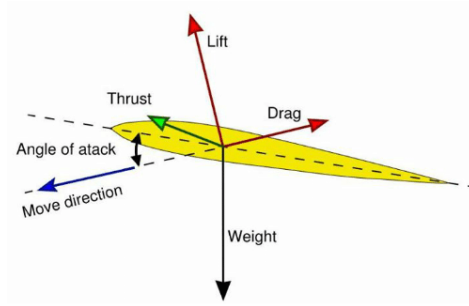


Why four rotors?



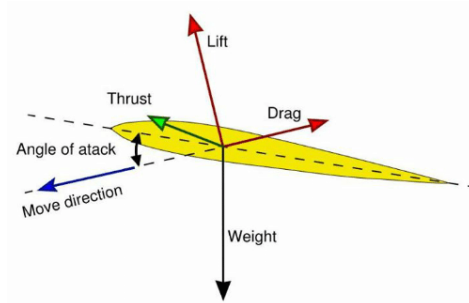
Aerodynamics

Thrust-Lift-Drag



Aerodynamics

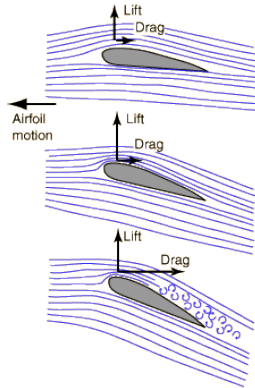
Thrust-Lift-Drag



Thrust, Lift and Drag are related to each other and to the design of the airfoil

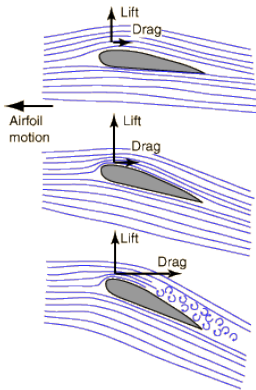
Aerodynamics

Angle of attack



Aerodynamics

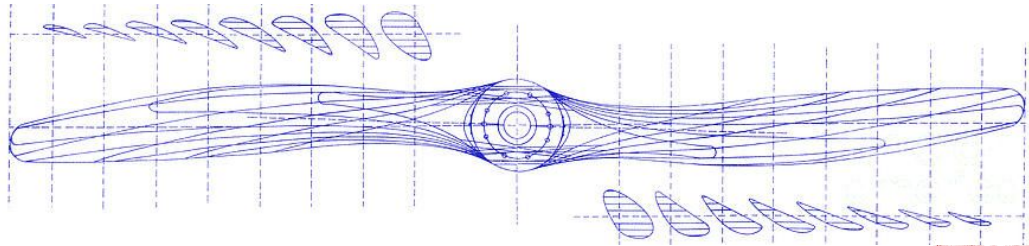
Angle of attack



The ratio of lift to drag are also related to the 'angle of attack'

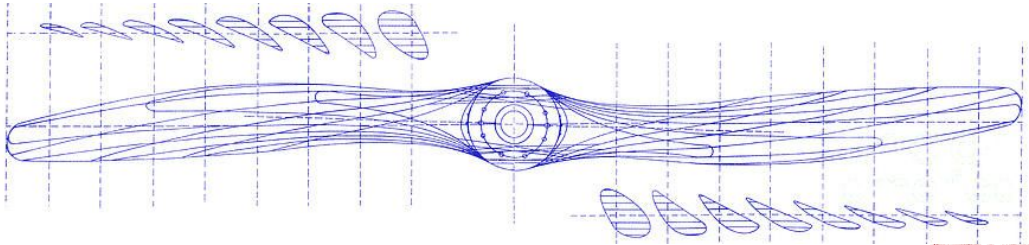
Aerodynamics

Propellers

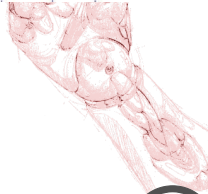


Aerodynamics

Propellers

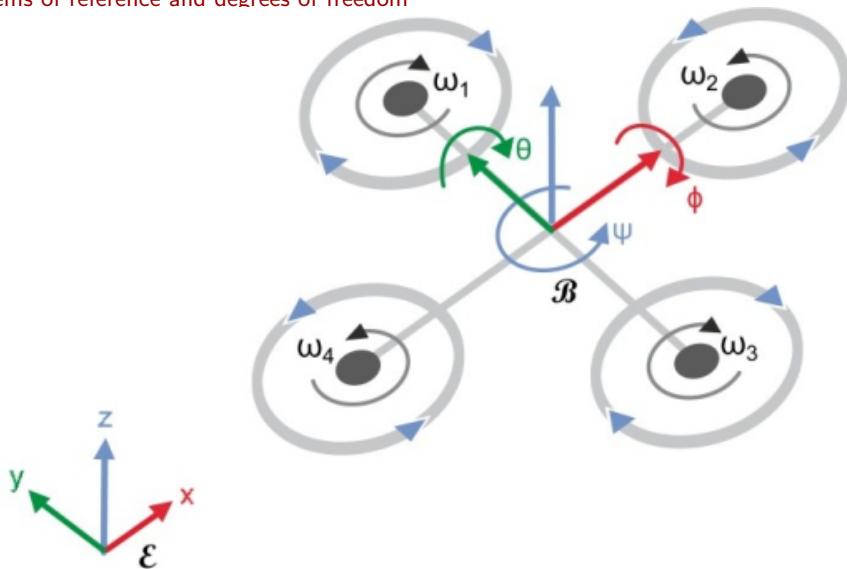


A propeller is many airfoils with different angles of attack



Quadrotor

Systems of reference and degrees of freedom



Quadrotor

Propeller forces and torques

Connecting with the aerodynamics of airfoils and propellers, each rotors produces a lifting force (F_i) and a torque (τ_i).



Quadrotor

Propeller forces and torques

Connecting with the aerodynamics of airfoils and propellers, each rotors produces a lifting force (F_i) and a torque (τ_i).

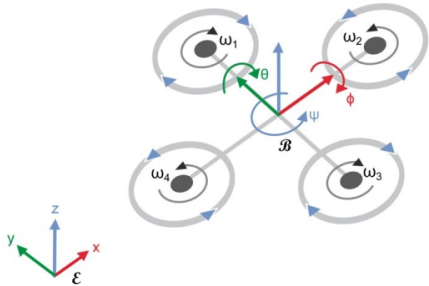
Both of these have to do with the design of the propeller, and are proportional to the square of the angular velocity of the propeller.

$$F_i = b\omega_i^2$$
$$\tau_i = d\omega_i^2$$



Quadrotor

Systems of reference and degrees of freedom

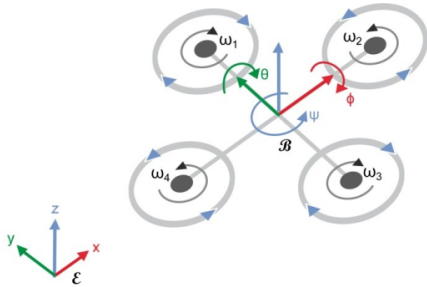


- x, y, z : translation along x, y, z axes of the world frame
- ϕ, θ, ψ : rotation around x, y, z axes of the drone frame



Quadrotor

Systems of reference and degrees of freedom



- x, y, z : translation along x, y, z axes of the world frame
- ϕ, θ, ψ : rotation around x, y, z axes of the drone frame

These are known as euler rotations (roll, pitch, yaw)

$$S = [\xi, \eta]^T, \text{ where: } \xi = [x, y, z]^T \text{ and } \eta = [\phi, \theta, \psi]^T$$



Kinematics modeling

Transformation

$$R_E^B(\eta) = \begin{bmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi s_\theta \\ c_\theta c_\psi & c_\phi c_\psi + s_\theta s_\phi s_\psi & c_\phi s_\theta s_\psi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}$$

where:

$$\begin{aligned} c_\theta &= \cos(\theta) \\ s_\psi &= \sin(\psi) \end{aligned}$$



Kinematics modeling

Transformation

$$R_E^B(\eta) = \begin{bmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi s_\theta \\ c_\theta c_\psi & c_\phi c_\psi + s_\theta s_\phi s_\psi & c_\phi s_\theta s_\psi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}$$

where:

$$c_\theta = \cos(\theta)$$

$$s_\psi = \sin(\psi)$$

Forward kinematics



Kinematics modeling

Transformation

$$R_E^B(\eta) = \begin{bmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi s_\theta \\ c_\theta c_\psi & c_\phi c_\psi + s_\theta s_\phi s_\psi & c_\phi s_\theta s_\psi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}$$

where:

$$\begin{aligned} c_\theta &= \cos(\theta) \\ s_\psi &= \sin(\psi) \end{aligned}$$

Forward kinematics

What is the difference compared to manipulators?



Quadrotor

Achieving flight

Quad-X Configuration

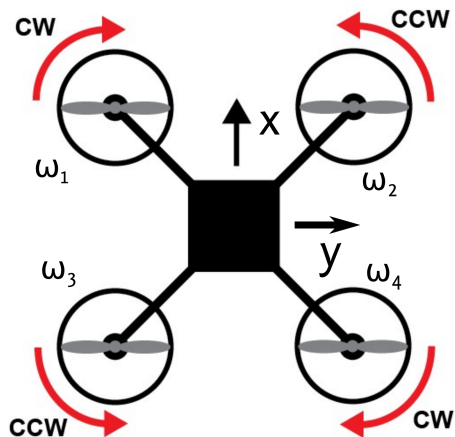


Figure 1

Quad-Plus Configuration

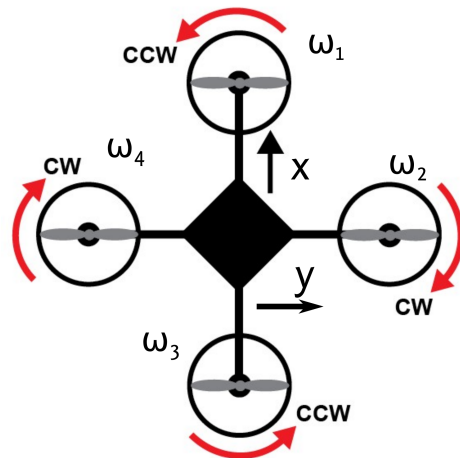


Figure 2

Quadrotor

Achieving flight in X configuration

Quad-X Configuration

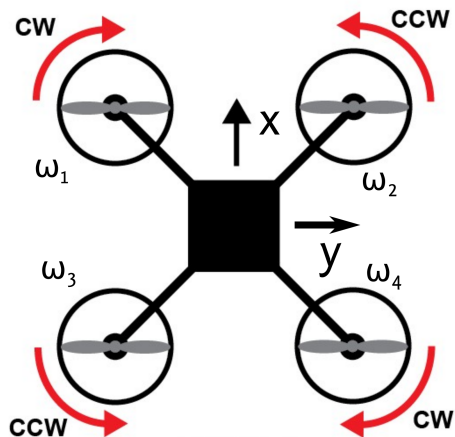


Figure 1



Quadrotor

Achieving flight in X configuration

Quad-X Configuration

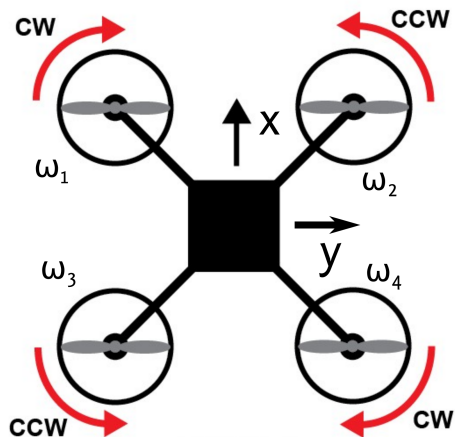


Figure 1

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$



Quadrotor

Achieving flight in X configuration

Quad-X Configuration

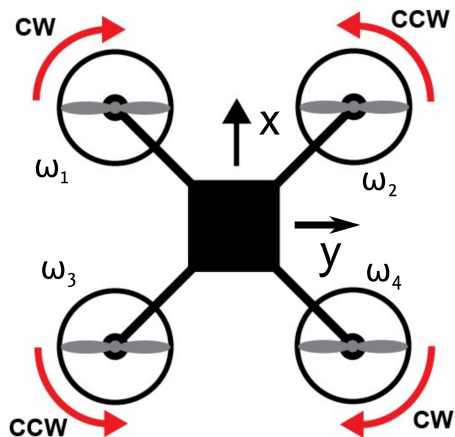


Figure 1

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 = \omega_4$



Quadrotor

Achieving flight in X configuration

Quad-X Configuration

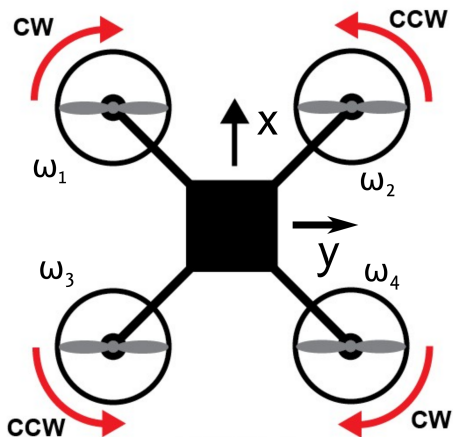


Figure 1

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 = \omega_4$
- Rotation around y: $\omega_1 = \omega_2, \omega_3 = \omega_4$



Quadrotor

Achieving flight in X configuration

Quad-X Configuration

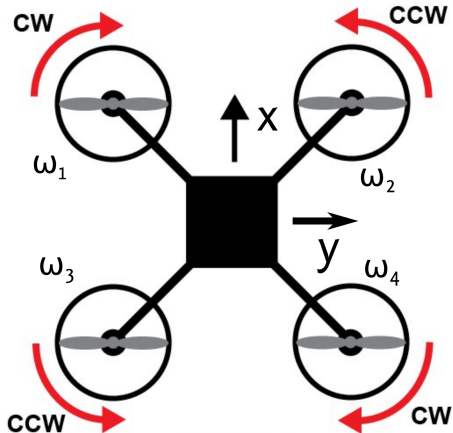


Figure 1

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 = \omega_4$
- Rotation around y: $\omega_1 = \omega_2, \omega_3 = \omega_4$
- Rotation around z:



Quadrotor

Achieving flight in X configuration

Quad-X Configuration

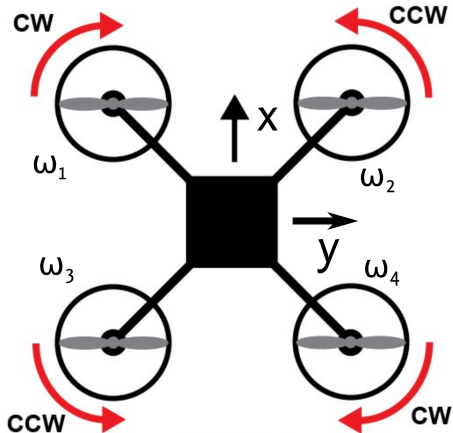
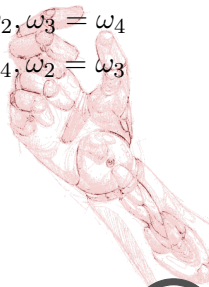


Figure 1

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 = \omega_4$
- Rotation around y: $\omega_1 = \omega_2, \omega_3 = \omega_4$
- Rotation around z: $\omega_1 = \omega_4, \omega_2 = \omega_3$



Quadrotor

Achieving flight in X configuration

Quad-X Configuration

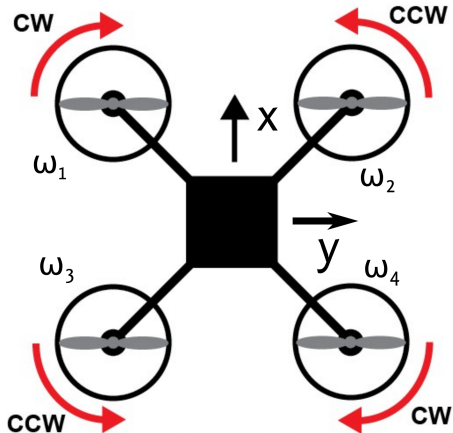
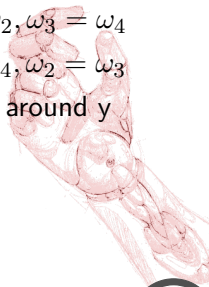


Figure 1

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 = \omega_4$
- Rotation around y: $\omega_1 = \omega_2, \omega_3 = \omega_4$
- Rotation around z: $\omega_1 = \omega_4, \omega_2 = \omega_3$
- Translation on x: Rotation around y



Quadrotor

Achieving flight in X configuration

Quad-X Configuration

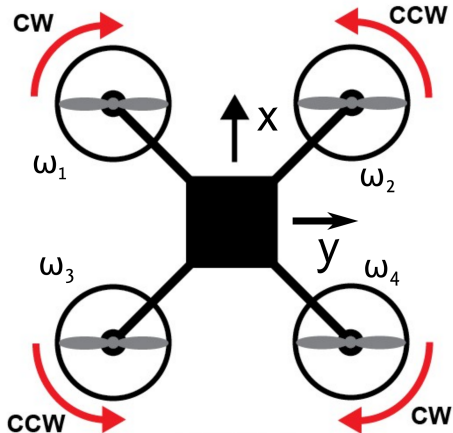


Figure 1

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 = \omega_4$
- Rotation around y: $\omega_1 = \omega_2, \omega_3 = \omega_4$
- Rotation around z: $\omega_1 = \omega_4, \omega_2 = \omega_3$
- Translation on x: Rotation around y
- Translation on y: Rotation around x

Quadrotor

Achieving flight in plus configuration

Quad-Plus Configuration

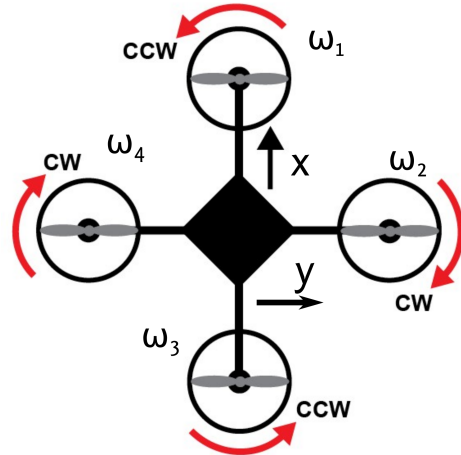


Figure 2

Quadrotor

Achieving flight in plus configuration

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$

Quad-Plus Configuration

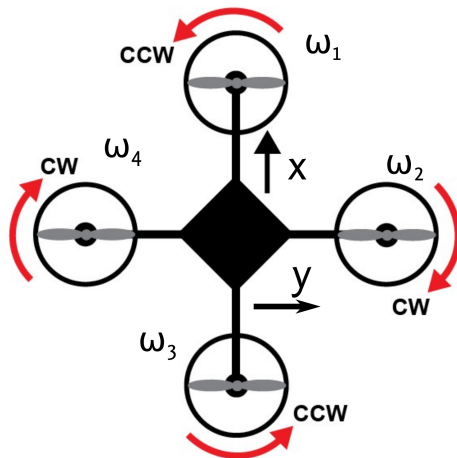


Figure 2

Quadrotor

Achieving flight in plus configuration

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 \neq \omega_4$

Quad-Plus Configuration

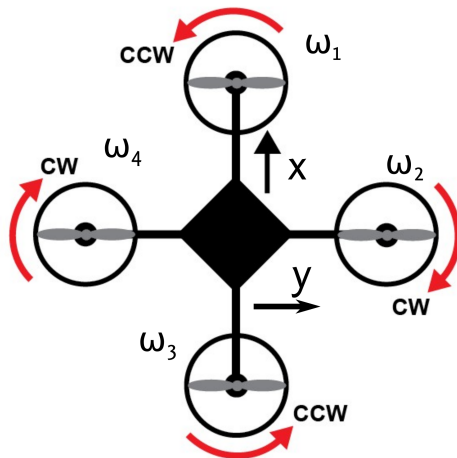


Figure 2

Quadrotor

Achieving flight in plus configuration

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 \neq \omega_4$
- Rotation around y: $\omega_1 \neq \omega_3, \omega_2 = \omega_4$

Quad-Plus Configuration

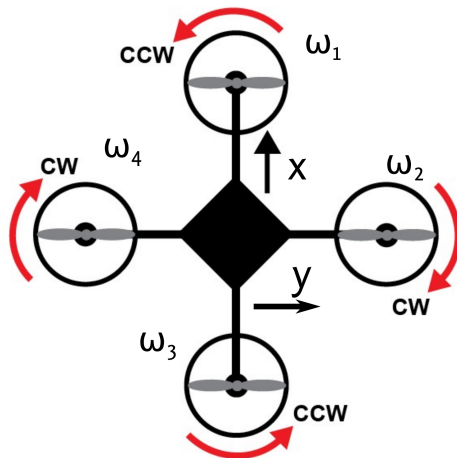


Figure 2

Quadrotor

Achieving flight in plus configuration

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 \neq \omega_4$
- Rotation around y: $\omega_1 \neq \omega_3, \omega_2 = \omega_4$
- Rotation around z: $\omega_1 = \omega_3, \omega_2 = \omega_4$

Quad-Plus Configuration

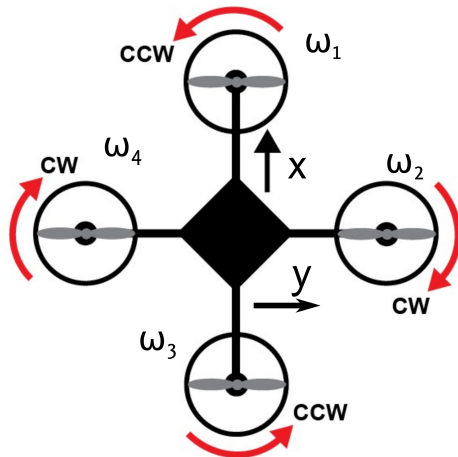


Figure 2

Quadrotor

Achieving flight in plus configuration

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 \neq \omega_4$
- Rotation around y: $\omega_1 \neq \omega_3, \omega_2 = \omega_4$
- Rotation around z: $\omega_1 = \omega_3, \omega_2 = \omega_4$
- Translation on x: Rotation around y

Quad-Plus Configuration

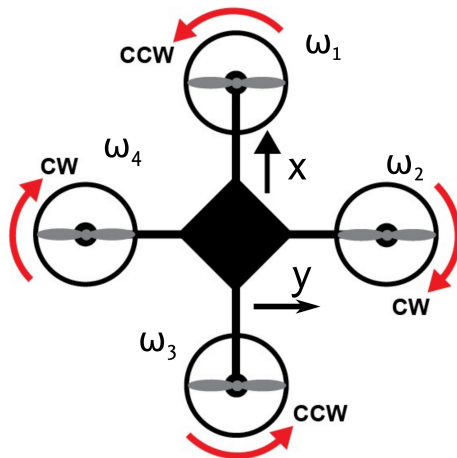


Figure 2

Quadrotor

Achieving flight in plus configuration

- Translation on z: $\omega_1 = \omega_2 = \omega_3 = \omega_4$
- Rotation around x: $\omega_1 = \omega_3, \omega_2 \neq \omega_4$
- Rotation around y: $\omega_1 \neq \omega_3, \omega_2 = \omega_4$
- Rotation around z: $\omega_1 = \omega_3, \omega_2 = \omega_4$
- Translation on x: Rotation around y
- Translation on y: Rotation around x

Quad-Plus Configuration

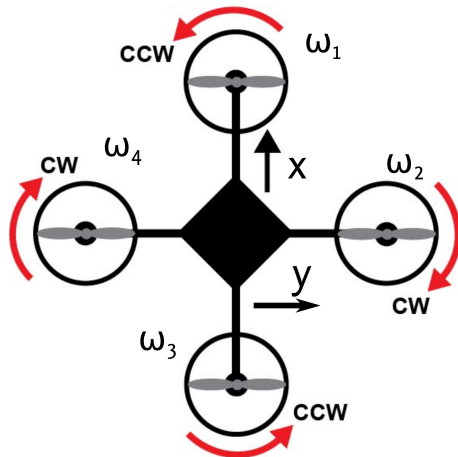


Figure 2

Dynamic modeling

Transformation

$$R_E^B(\eta) = \begin{bmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi s_\theta \\ c_\theta c_\psi & c_\phi c_\psi + s_\theta s_\phi s_\psi & c_\phi s_\theta s_\psi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}$$

Forward kinematics



Dynamic modeling

Transformation

$$R_E^B(\eta) = \begin{bmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi s_\theta \\ c_\theta c_\psi & c_\phi c_\psi + s_\theta s_\phi s_\psi & c_\phi s_\theta s_\psi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}$$

Forward kinematics

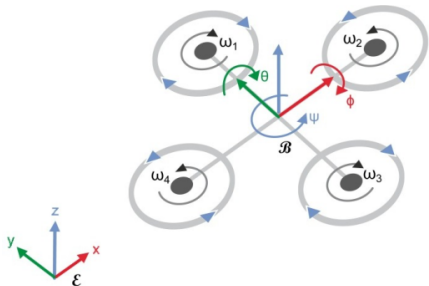
What about inverse kinematics?



Dynamic modeling

The Lagrangian

Remember:



- x, y, z : translation along x, y, z axes of the fixed frame
- ϕ, θ, ψ : rotation around x, y, z axes of the fixed frame

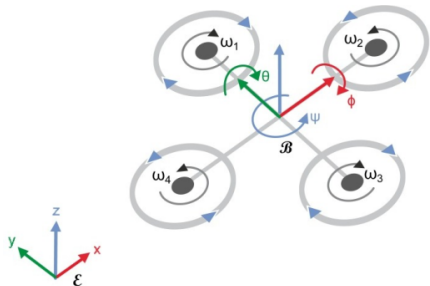
$$S = [\xi, \eta]^T, \text{ where: } \xi = [x, y, z]^T \text{ and } \eta = [\phi, \theta, \psi]^T$$



Dynamic modeling

The Lagrangian

Remember:



- x, y, z : translation along x, y, z axes of the fixed frame
- ϕ, θ, ψ : rotation around x, y, z axes of the fixed frame

$$S = [\xi, \eta]^T, \text{ where: } \xi = [x, y, z]^T \text{ and } \eta = [\phi, \theta, \psi]^T$$

Therefore:

$$L(S, \dot{S}) = K_{lin} + K_{rot} - P = \frac{1}{2} \left(m \dot{\xi}^T \dot{\xi} + \dot{\eta}^T I(\eta) \dot{\eta} \right) - mgz$$



Dynamic modeling

The Lagrangian

$$L(S, \dot{S}) = K_{lin} + K_{rot} - P = \frac{1}{2} \left(m \dot{\xi}^T \dot{\xi} + \dot{\eta}^T I(\eta) \dot{\eta} \right) - mgz$$

Where $I(\eta)$ is the moments of inertia expressed in the fixed frame.



Dynamic modeling

The Lagrangian

$$L(S, \dot{S}) = K_{lin} + K_{rot} - P = \frac{1}{2} \left(m \dot{\xi}^T \dot{\xi} + \dot{\eta}^T I(\eta) \dot{\eta} \right) - mgz$$

Where $I(\eta)$ is the moments of inertia expressed in the fixed frame.

By differentiating, we can calculate the equation of motion of the quadrotor



Dynamic modeling

The Lagrangian

$$L(S, \dot{S}) = K_{lin} + K_{rot} - P = \frac{1}{2} \left(m \dot{\xi}^T \dot{\xi} + \dot{\eta}^T I(\eta) \dot{\eta} \right) - mgz$$

Where $I(\eta)$ is the moments of inertia expressed in the fixed frame.

By differentiating, we can calculate the equation of motion of the quadrotor

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{S}} - \frac{\partial L}{\partial S} = \begin{bmatrix} F_{transl} \\ F_{rot} \end{bmatrix}$$



Dynamic modeling

The Lagrangian

Since the translation kinetic and potential energies depend only on ξ and $\dot{\xi}$ and the rotational kinetic energy only on $\dot{\eta}$, we can break this down to two decoupled systems of equations.

$$\frac{d}{dt} \frac{\partial (K_{transl} + P)}{\partial \dot{\xi}} - \frac{\partial (K_{transl} + P)}{\partial \xi} = F_{transl}$$

$$\frac{d}{dt} \frac{\partial K_{rot}}{\partial \dot{\eta}} - \frac{\partial K_{rot}}{\partial \eta} = F_{rot}$$



Dynamic modeling

The Lagrangian

$$L(S, \dot{S}) = \frac{1}{2} \left(m \dot{\xi}^T \dot{\xi} + \dot{\eta}^T I(\eta) \dot{\eta} \right) - mgz$$

The first equation is easy to differentiate:



Dynamic modeling

The Lagrangian

$$L(S, \dot{S}) = \frac{1}{2} \left(m \dot{\xi}^T \dot{\xi} + \dot{\eta}^T I(\eta) \dot{\eta} \right) - mgz$$

The first equation is easy to differentiate:

$$F_{transl} = m \ddot{\xi} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$



Dynamic modeling

The Lagrangian

$$L(S, \dot{S}) = \frac{1}{2} \left(m \dot{\xi}^T \dot{\xi} + \dot{\eta}^T I(\eta) \dot{\eta} \right) - mgz$$

The first equation is easy to differentiate:

$$F_{transl} = m \ddot{\xi} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

The second one is a bit more 'stiff', due to the moments of inertia:



Dynamic modeling

The Lagrangian

$$L(S, \dot{S}) = \frac{1}{2} \left(m \dot{\xi}^T \dot{\xi} + \dot{\eta}^T I(\eta) \dot{\eta} \right) - mgz$$

The first equation is easy to differentiate:

$$F_{transl} = m \ddot{\xi} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

The second one is a bit more 'stiff', due to the moments of inertia:

$$F_{rot} = I(\eta) \ddot{\eta} + \dot{I}(\eta) \dot{\eta} - \frac{1}{2} \frac{d}{d\eta} \left(\dot{\eta}^T I(\eta) \dot{\eta} \right) = I(\eta) \ddot{\eta} + C(\eta, \dot{\eta}) \dot{\eta}$$



Dynamic modeling

The forces

If we consider a hover flight, with very small changes in orientation, we need to consider a vertical force to achieve this:

$$F_{transl} = \begin{bmatrix} 0 \\ 0 \\ U_z \end{bmatrix}$$



Dynamic modeling

The forces

If we consider a hover flight, with very small changes in orientation, we need to consider a vertical force to achieve this:

$$F_{transl} = \begin{bmatrix} 0 \\ 0 \\ U_z \end{bmatrix}$$

We need to express this force on the frame attached on the drone, therefore:

$$F_{transl} = R_E^B \begin{bmatrix} 0 \\ 0 \\ U_z \end{bmatrix}$$



Dynamic modeling

The forces

If we do the multiplication between force and transformation matrix, we get:

$$\begin{bmatrix} c_\theta c_\psi & c_\psi s_\theta s_\phi - c_\phi s_\psi & s_\phi s_\psi + c_\phi c_\psi s_\theta \\ c_\theta c_\psi & c_\phi c_\psi + s_\theta s_\phi s_\psi & c_\phi s_\theta s_\psi - c_\psi s_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ U_z \end{bmatrix} = m\ddot{\xi} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

$$\begin{aligned} \ddot{x} &= \left[s_\phi s_\psi + c_\phi c_\psi s_\theta \right] \frac{U_z}{m} \\ \ddot{y} &= \left[c_\phi s_\theta s_\psi - c_\psi s_\phi \right] \frac{U_z}{m} \\ \ddot{z} &= -g + c_\phi c_\theta \frac{U_z}{m} \end{aligned}$$



Dynamic modeling

The torques

Since we usually express rotations on the frame attached on the drone, the rotational forces (torques), do not need to be 'transformed' in the drone's body frame. Therefore:

$$F_{rot} = \begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix}$$



Dynamic modeling

The torques

Since we usually express rotations on the frame attached on the drone, the rotational forces (torques), do not need to be 'transformed' in the drone's body frame. Therefore:

$$F_{rot} = \begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix} = I(\eta)\ddot{\eta} + C(\eta, \dot{\eta})\dot{\eta}$$



Dynamic modeling

The torques

Since we usually express rotations on the frame attached on the drone, the rotational forces (torques), do not need to be 'transformed' in the drone's body frame. Therefore:

$$F_{rot} = \begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix} = I(\eta)\ddot{\eta} + C(\eta, \dot{\eta})\dot{\eta}$$

$$\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = I^{-1}(\eta) \left(\begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix} - C(\eta, \dot{\eta})\dot{\eta} \right)$$



Dynamic modeling

Putting it all together

And all together:

$$\begin{aligned}\ddot{x} &= \left[c_\phi s_\theta c_\psi + s_\phi s_\psi \right] \frac{U_z}{m} \\ \ddot{y} &= \left[c_\phi s_\theta s_\psi - s_\phi c_\psi \right] \frac{U_z}{m} \\ \ddot{z} &= -g + c_\phi c_\theta \frac{U_z}{m} \\ \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} &= I^{-1}(\eta) \left(\begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix} - C(\eta, \dot{\eta})\dot{\eta} \right)\end{aligned}$$



Dynamic modeling

Simplifications

Since we consider the hovering motion, this means that the rotations are small and very slow, therefore we have:

$$\cos(\alpha) \approx 1$$

$$\sin(\alpha) \approx \alpha$$

$$\dot{\eta} \approx 0$$



Dynamic modeling

Simplifications

Since we consider the hovering motion, this means that the rotations are small and very slow, therefore we have:

$$\cos(\alpha) \approx 1$$

$$\sin(\alpha) \approx \alpha$$

$$\dot{\eta} \approx 0$$

The upward force, can be also expressed as $U_z = mg + \Delta U_z$



Dynamic modeling

Simplifications

Since we consider the hovering motion, this means that the rotations are small and very slow, therefore we have:

$$\begin{aligned}\cos(\alpha) &\approx 1 \\ \sin(\alpha) &\approx \alpha \\ \dot{\eta} &\approx 0\end{aligned}$$

The upward force, can be also expressed as $U_z = mg + \Delta U_z$ Therefore, our equations of motion become:

$$\begin{aligned}\ddot{x} &= \theta g \\ \ddot{y} &= -\phi g \\ \ddot{z} &= \frac{\Delta U_z}{m}\end{aligned}$$

$$\begin{aligned}\ddot{\phi} &= \frac{1}{I_{xx}} U_{\phi} \\ \ddot{\theta} &= \frac{1}{I_{yy}} U_{\theta} \\ \ddot{\psi} &= \frac{1}{I_{zz}} U_{\psi}\end{aligned}$$



Dynamic modeling

What are the control inputs?

We need to consider again how does a drone achieve flight. For the plus configuration:

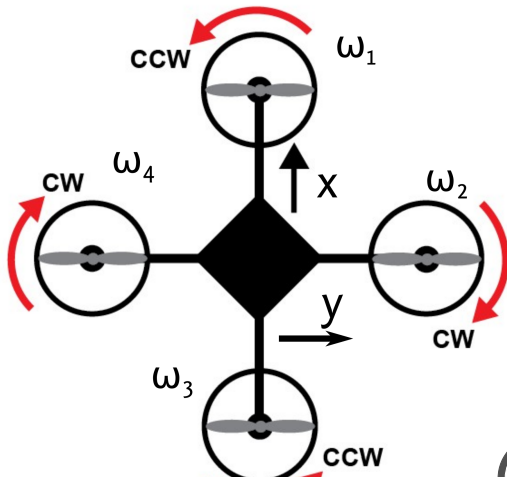
Quad-Plus Configuration

$$U_z = F_1 + F_2 + F_3 + F_4$$

$$U_\phi = l(F_4 - F_2)$$

$$U_\theta = l(F_1 - F_3)$$

$$U_\psi = \tau_1 + \tau_3 - \tau_2 - \tau_4$$



Dynamic modeling

What are the control inputs?

We need to consider again how does a drone achieve flight. For the plus configuration:

Quad-Plus Configuration

$$U_z = F_1 + F_2 + F_3 + F_4$$

$$U_\phi = l(F_4 - F_2)$$

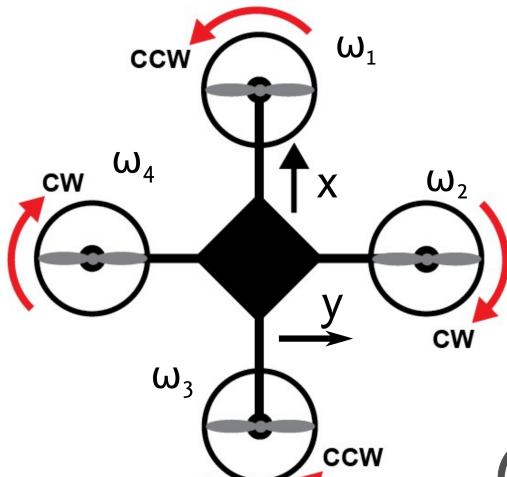
$$U_\theta = l(F_1 - F_3)$$

$$U_\psi = \tau_1 + \tau_3 - \tau_2 - \tau_4$$

Remember:

$$F_i = b\omega_i^2$$

$$\tau_i = d\omega_i^2$$

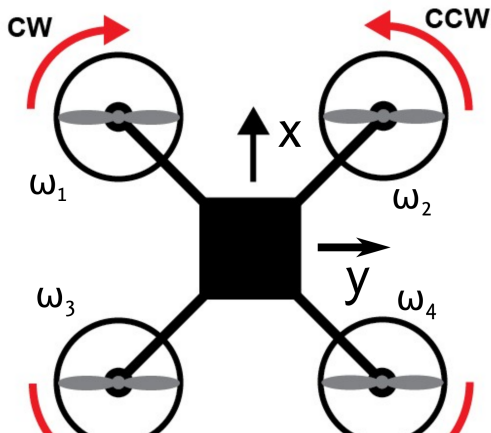


Dynamic modeling

What are the control inputs?

For the cross configurations

Quad-X Configuration



$$U_z = F_1 + F_2 + F_3 + F_4$$

$$U_\phi = \frac{\sqrt{2}}{2}l(F_1 + F_3 - F_2 - F_4)$$

$$U_\theta = \frac{\sqrt{2}}{2}l(F_1 + F_2 - F_3 - F_4)$$

$$U_\psi = \tau_2 + \tau_3 - \tau_1 - \tau_4$$

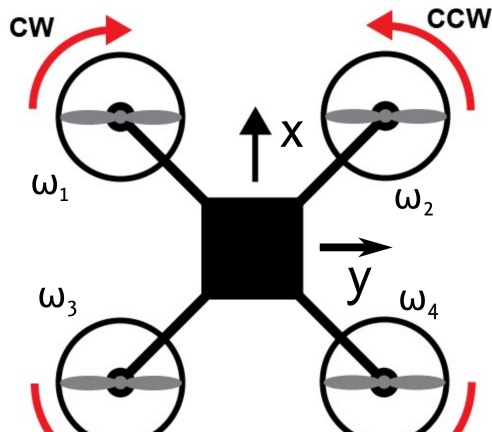


Dynamic modeling

What are the control inputs?

For the cross configurations

Quad-X Configuration



$$U_z = F_1 + F_2 + F_3 + F_4$$

$$U_\phi = \frac{\sqrt{2}}{2}l(F_1 + F_3 - F_2 - F_4)$$

$$U_\theta = \frac{\sqrt{2}}{2}l(F_1 + F_2 - F_3 - F_4)$$

$$U_\psi = \tau_2 + \tau_3 - \tau_1 - \tau_4$$

Is this useful?



Quadrotor

Control

If we can derive the dynamic model of a quadrotor, why do you think control is difficult?



Quadrotor

Control

If we can derive the dynamic model of a quadrotor, why do you think control is difficult?

- We made several assumptions to end up with a linear model



Quadrotor

Control

If we can derive the dynamic model of a quadrotor, why do you think control is difficult?

- We made several assumptions to end up with a linear model
- We did not include motor dynamics



Quadrotor

Control

If we can derive the dynamic model of a quadrotor, why do you think control is difficult?

- We made several assumptions to end up with a linear model
- We did not include motor dynamics
- We did not include aerodynamics



Quadrotor

Control

If we can derive the dynamic model of a quadrotor, why do you think control is difficult?

- We made several assumptions to end up with a linear model
- We did not include motor dynamics
- We did not include aerodynamics
- When flying outdoors, there are huge disturbances (wind)



Quadrotor

Control

If we can derive the dynamic model of a quadrotor, why do you think control is difficult?

- We made several assumptions to end up with a linear model
- We did not include motor dynamics
- We did not include aerodynamics
- When flying outdoors, there are huge disturbances (wind)
- A quadrotor is underactuated



Quadrotor

Control

If we can derive the dynamic model of a quadrotor, why do you think control is difficult?

- We made several assumptions to end up with a linear model
- We did not include motor dynamics
- We did not include aerodynamics
- When flying outdoors, there are huge disturbances (wind)
- A quadrotor is underactuated

For simple slow motions though, we can try to control it with this simple model



Quadrotor

Control

The model we have calculated is already linearized, so we can apply state feedback control. Our states and inputs are:

$$x = [\xi, \dot{\xi}]^T$$

$$u = [U_z, U_\phi, U_\theta, U_\psi]^T$$

The model can be written in state space as:

$$\dot{x} = Ax + Bu$$



Quadrotor

State feedback

By calculating the matrices **A** and **B**, we can calculate a feedback gain matrix **K**, that will stabilize the system. We do that using standard pole positioning. Our system has 12 states and 4 inputs, so the gain matrix will have dimensions 4×12 :

$$u = -Kx$$



Quadrotor

State feedback

By calculating the matrices **A** and **B**, we can calculate a feedback gain matrix **K**, that will stabilize the system. We do that using standard pole positioning. Our system has 12 states and 4 inputs, so the gain matrix will have dimensions 4×12 :

$$u = -Kx$$

We then add an integrator to track a specific setpoint, in our case a position in space:

$$u = -Kx + u_o$$
$$u_o = [u_z, u_y, u_x, 0]^T$$

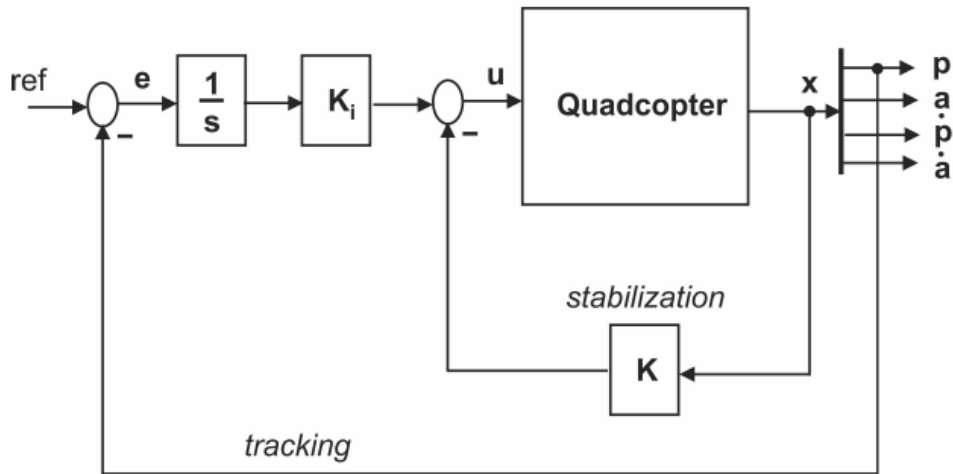
where:

$$u_x = \frac{ki_x}{s} (r_x - x), u_y = \frac{ki_y}{s} (r_y - y), u_z = \frac{ki_z}{s} (r_z - z)$$



Quadrotor

State feedback



Further reading/watching

Very cool TED talk on drone modeling:

<https://www.youtube.com/watch?v=w2itwFJCgFQ>





Questions?