Drone modeling

Kinematics, Aerodynamics, Dynamics, Control

TECHNICAL UNIVERSITY OF CLUJ-NAPOCA ROMANIA

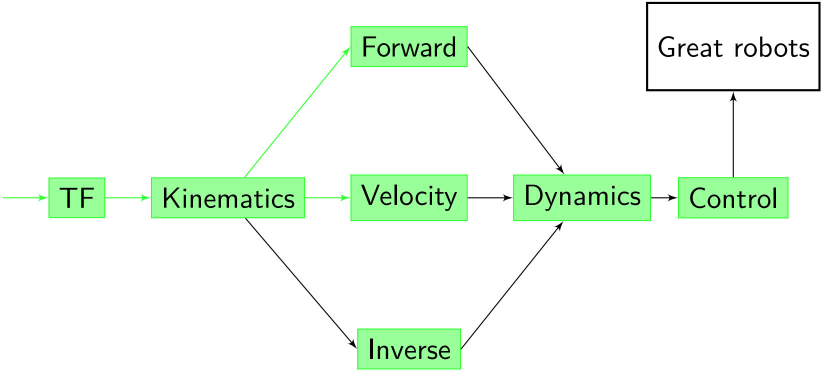
Last update: December 9, 2024

Agenda

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

Grand scheme

The big picture



Quadrotor drones

What is a quadrotor?

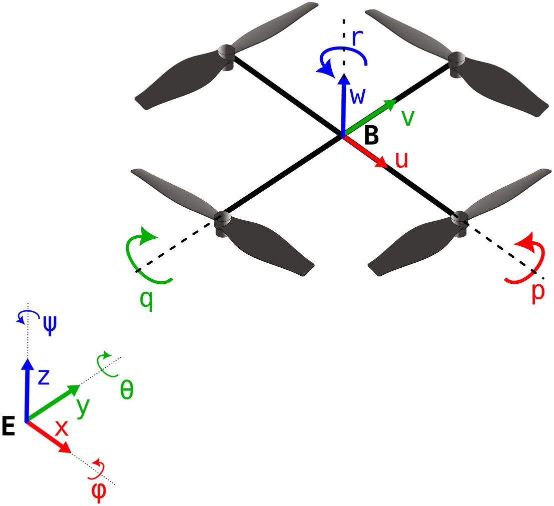
Quadrotor drones

What is a quadrotor?

Why four rotors?

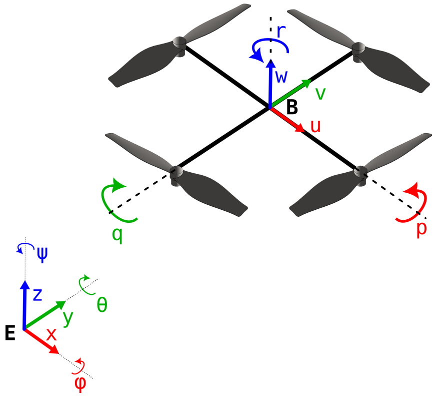
Quadrotor

Systems of reference and degrees of freedom



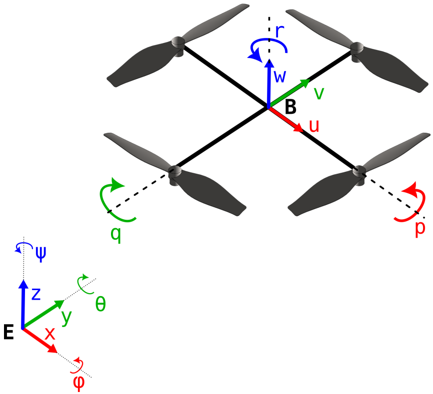
Quadrotor

Systems of reference and degrees of freedom



Quadrotor

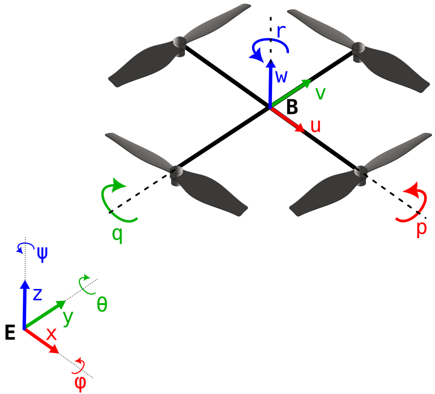
Systems of reference and degrees of freedom



- x, y, z: translation along principal axes of the earth frame

Quadrotor

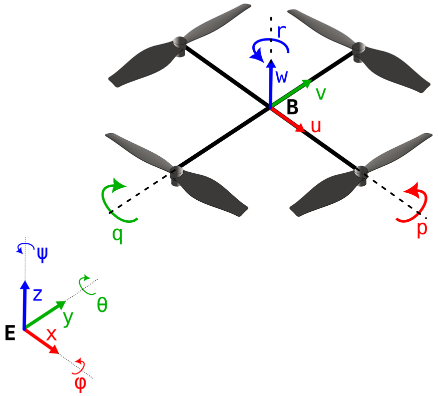
Systems of reference and degrees of freedom



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

Quadrotor

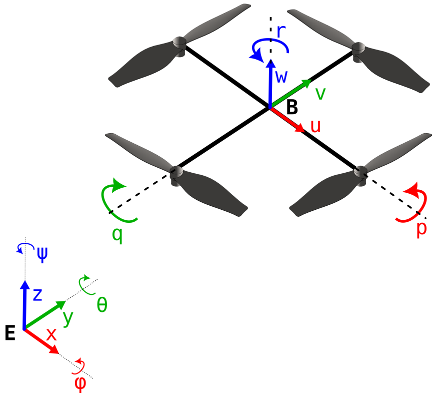
Systems of reference and degrees of freedom



- x, y, z: translation along principal axes of the earth frame
- ϕ, θ, ψ: rotation around principal axes of the earth frame
- u, v, w: velocity along principal axes of the drone body frame

Quadrotor

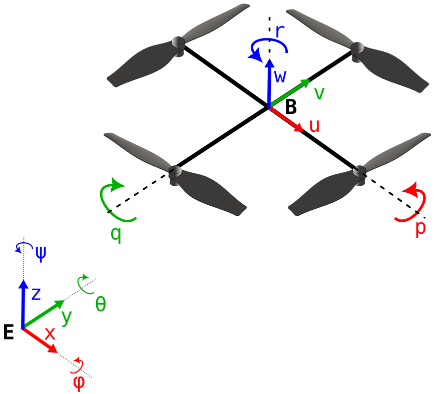
Systems of reference and degrees of freedom



- x, y, z: translation along principal axes of the earth frame
- ϕ, θ, ψ: rotation around principal axes of the earth frame
- u, v, w: velocity along principal axes of the drone body frame
- p, q, r: angular velocity around principal axes of the drone body frame

Quadrotor

Systems of reference and degrees of freedom

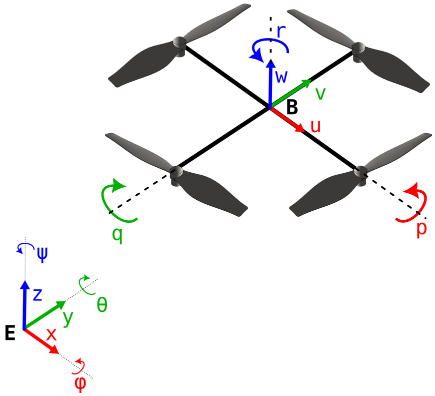


- x, y, z: translation along principal axes of the earth frame
- ϕ, θ, ψ: rotation around principal axes of the earth frame
- u, v, w: velocity along principal axes of the drone body frame
- p, q, r: angular velocity around principal axes of the drone body frame

Why velocities and not positions for the drone body frame?

Quadrotor

Systems of reference and degrees of freedom



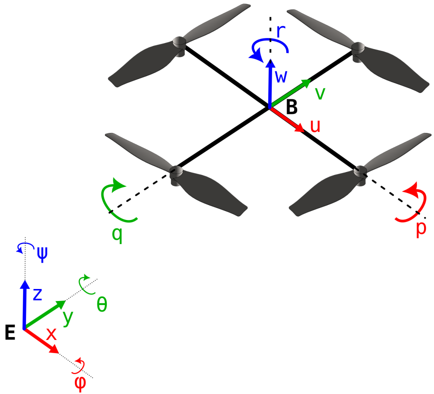
- x, y, z: translation along principal axes of the earth frame
- ϕ, θ, ψ: rotation around principal axes of the earth frame
- u, v, w: velocity along principal axes of the drone body frame
- p, q, r: angular velocity around principal axes of the drone body frame

Why velocities and not positions for the drone body frame?

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

Quadrotor

Systems of reference and degrees of freedom



- x, y, z: translation along principal axes of the earth frame
- ϕ, θ, ψ: rotation around principal axes of the earth frame
- u, v, w: velocity along principal axes of the drone body frame
- p, q, r: angular velocity around principal axes of the drone body frame

Why velocities and not positions for the drone body frame?

$P = [\xi, \eta]^T$, where: $\xi = [x, y, z]^T$ and $\eta = [\phi, \theta, \psi]^T$
$V = [\pi, \rho]^T$, where: $\pi = [u, v, p]^T$ and $\rho = [p, q, r]^T$

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:

$$c_\theta = cos(\theta)$$
$$s_\psi = sin(\psi)$$

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:

$$c_\theta = cos(\theta)$$
$$s_\psi = sin(\psi)$$

We can use this to convert velocities $\dot{\xi} = R_E^B(\eta)\pi$

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:

$$c_\theta = cos(\theta)$$
$$s_\psi = sin(\psi)$$

We can use this to convert velocities $\dot{\xi} = R_E^B(\eta)\pi$

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:

$$c_\theta = cos(\theta)$$
$$s_\psi = sin(\psi)$$

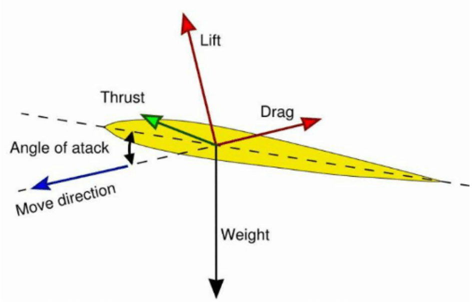
We can use this to convert velocities $\dot{\xi} = R_E^B(\eta)\pi$

Forward kinematics

What about inverse kinematics?

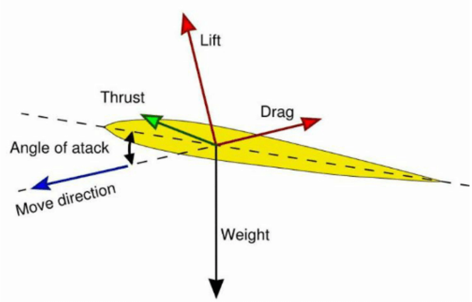
Aerodynamics

Thrust-Lift-Drag



Aerodynamics

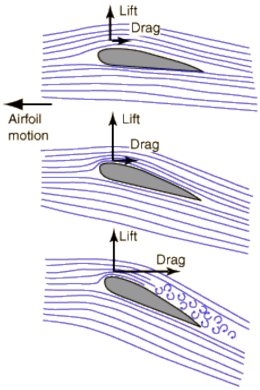
Thrust-Lift-Drag



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

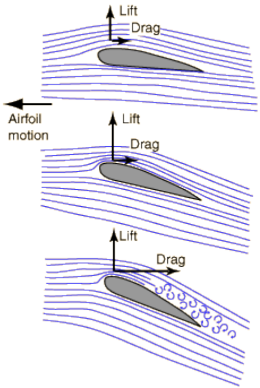
Aerodynamics

Angle of attack



Aerodynamics

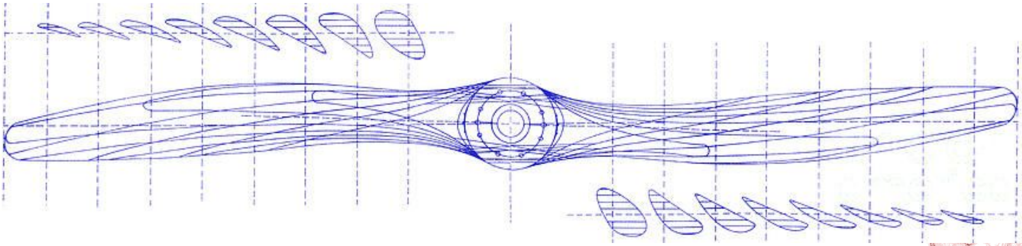
Angle of attack



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

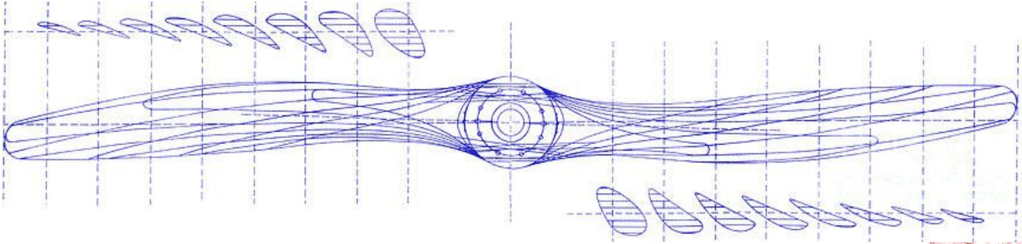
Aerodynamics

Propellers



Aerodynamics

Propellers

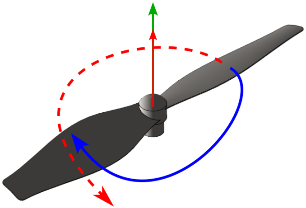


A propeller is many airfoils with different angles of attack

Quadrotor

Propeller forces and torques

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



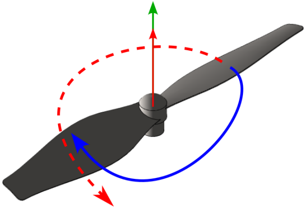
Quadrotor

Propeller forces and torques

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

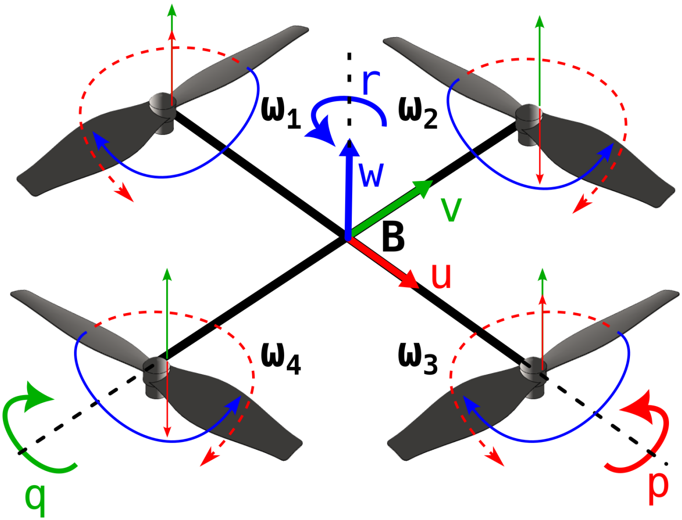
Both of these have to do with the design of the propeller, and are proportional to the square of the angular velocity of the propeller.

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



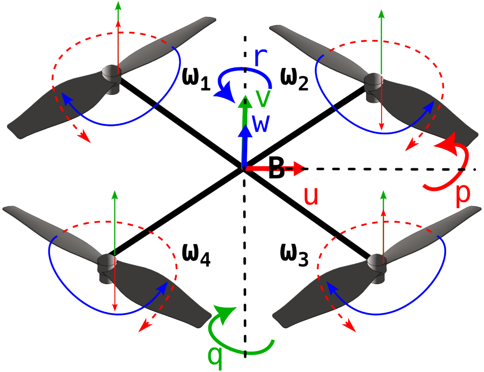
Quadrotor

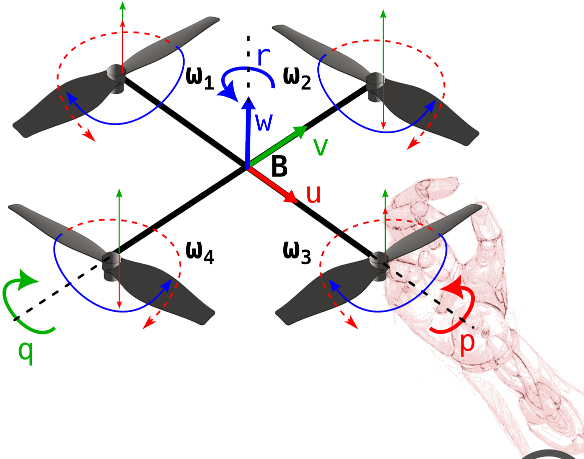
Achieving flight



Quadrotor

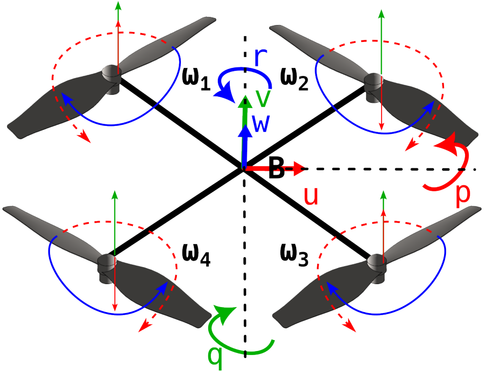
Achieving flight





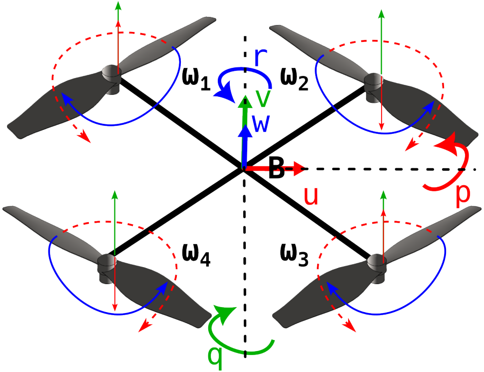
Quadrotor

Achieving flight in X configuration



Quadrotor

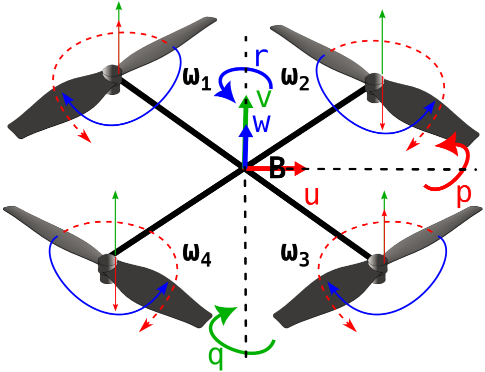
Achieving flight in X configuration



- $\dot{w}: \omega_1 = \omega_2 = \omega_3 = \omega_4$

Quadrotor

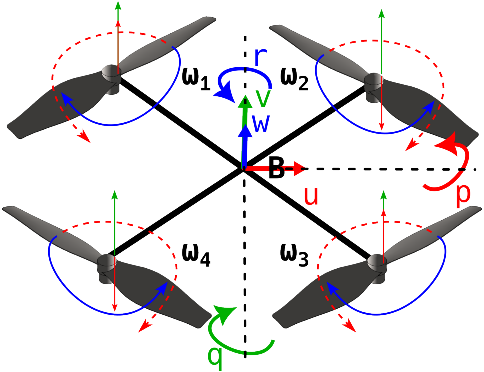
Achieving flight in X configuration



- $\dot{w}: \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p}: \omega_1 = \omega_2 \neq \omega_3 = \omega_4$

Quadrotor

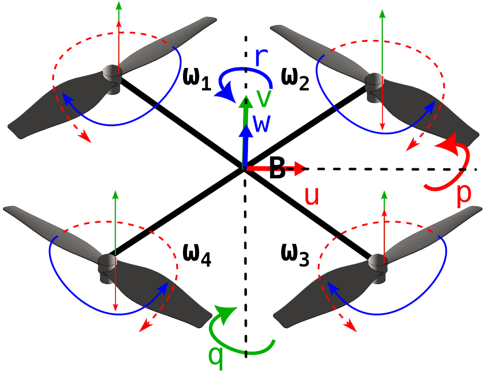
Achieving flight in X configuration



- $\dot{w} : \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p} : \omega_1 = \omega_2 \neq \omega_3 = \omega_4$
- $\dot{q} : \omega_1 = \omega_4 \neq \omega_2 = \omega_3$

Quadrotor

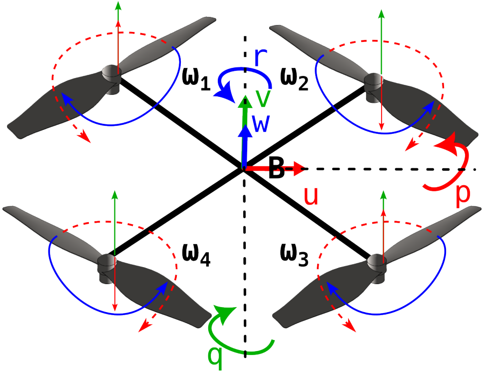
Achieving flight in X configuration



- $\dot{w}: \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p}: \omega_1 = \omega_2 \neq \omega_3 = \omega_4$
- $\dot{q}: \omega_1 = \omega_4 \neq \omega_2 = \omega_3$
- $\dot{r}$:

Quadrotor

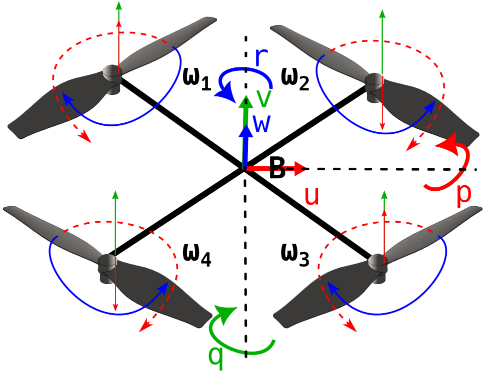
Achieving flight in X configuration



- $\dot{w}: \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p}: \omega_1 = \omega_2 \neq \omega_3 = \omega_4$
- $\dot{q}: \omega_1 = \omega_4 \neq \omega_2 = \omega_3$
- $\dot{r}: \omega_1 = \omega_3 \neq \omega_2 = \omega_4$

Quadrotor

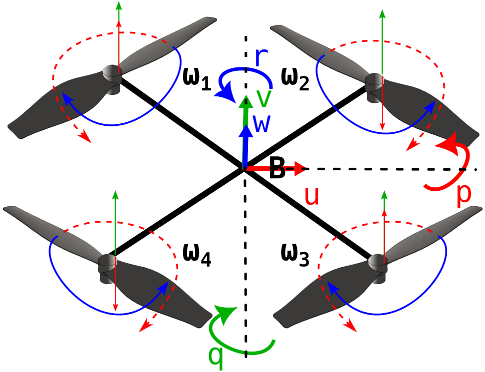
Achieving flight in X configuration



- $\dot{w}: \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p}: \omega_1 = \omega_2 \neq \omega_3 = \omega_4$
- $\dot{q}: \omega_1 = \omega_4 \neq \omega_2 = \omega_3$
- $\dot{r}: \omega_1 = \omega_3 \neq \omega_2 = \omega_4$
- $\dot{u} = \dot{q}$

Quadrotor

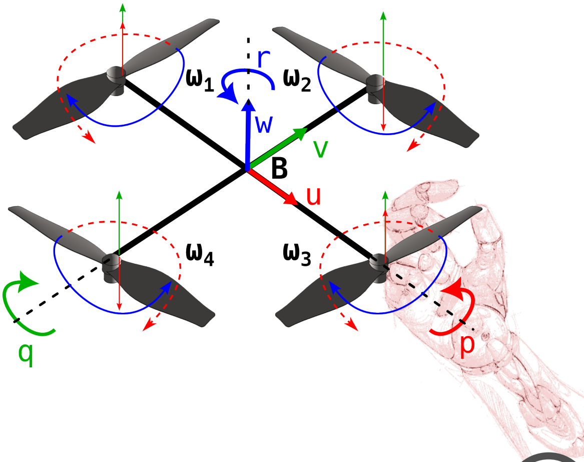
Achieving flight in X configuration



- $\dot{w} : \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p} : \omega_1 = \omega_2 \neq \omega_3 = \omega_4$
- $\dot{q} : \omega_1 = \omega_4 \neq \omega_2 = \omega_3$
- $\dot{r}$: $\omega_1 = \omega_3 \neq \omega_2 = \omega_4$
- $\dot{u} = \dot{q}$
- $\dot{v} = \dot{p}$

Quadrotor

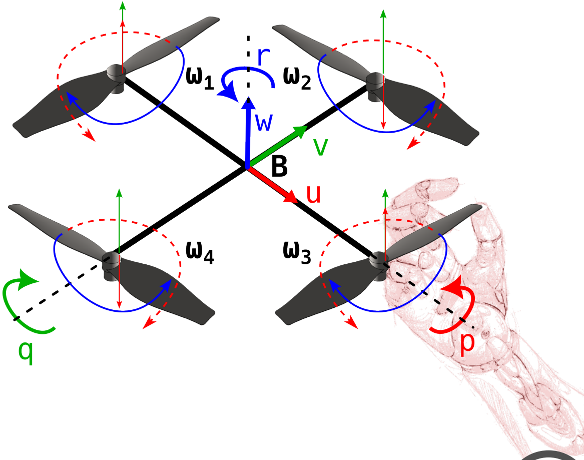
Achieving flight in plus configuration



Quadrotor

Achieving flight in plus configuration

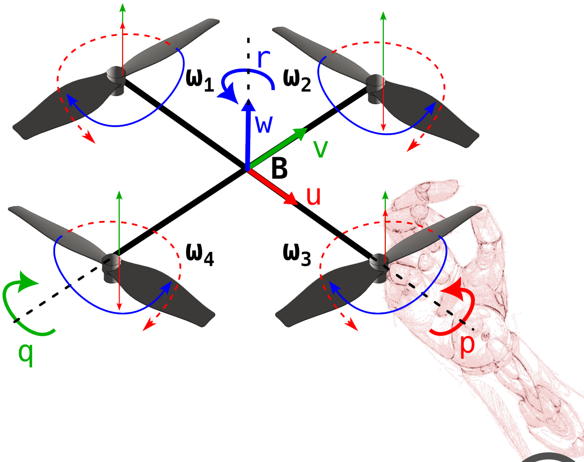
- $\dot{w} : \omega_1 = \omega_2 = \omega_3 = \omega_4$



Quadrotor

Achieving flight in plus configuration

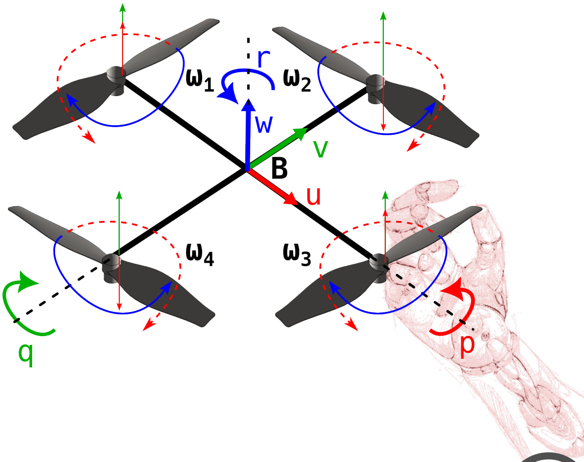
- $\dot{w} : \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p} : \omega_1 = \omega_3, \omega_2 \neq \omega_4$



Quadrotor

Achieving flight in plus configuration

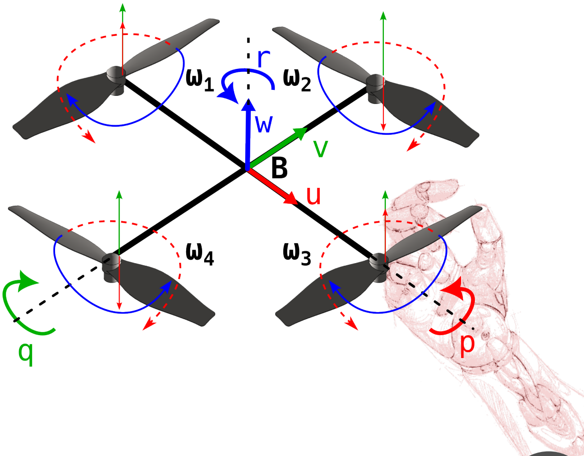
- $\dot{w} : \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p} : \omega_1 = \omega_3, \omega_2 \neq \omega_4$
- $\dot{q} : \omega_1 \neq \omega_3, \omega_2 = \omega_4$



Quadrotor

Achieving flight in plus configuration

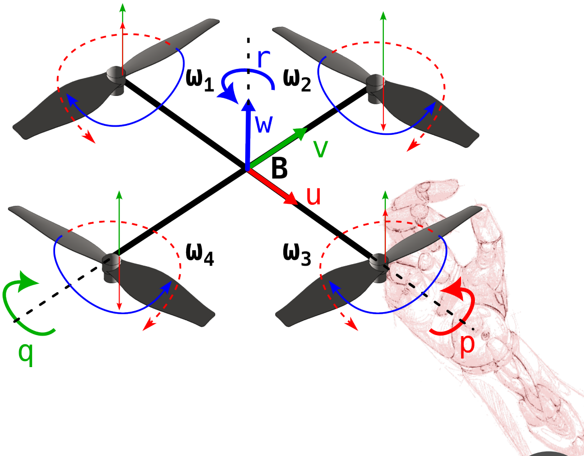
- $\dot{w} : \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p} : \omega_1 = \omega_3, \omega_2 \neq \omega_4$
- $\dot{q} : \omega_1 \neq \omega_3, \omega_2 = \omega_4$
- $\dot{r} : \omega_1 = \omega_3 \neq \omega_2 = \omega_4$



Quadrotor

Achieving flight in plus configuration

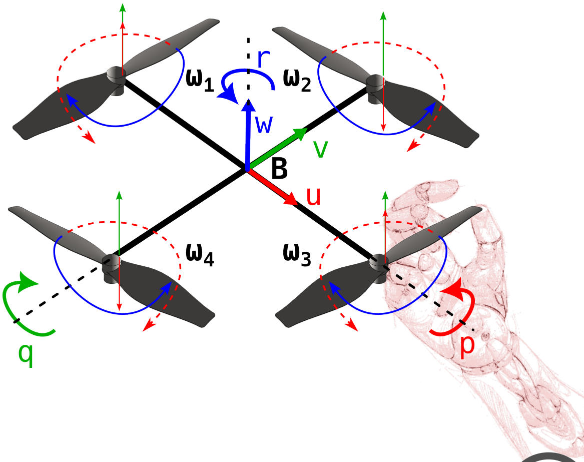
- $\dot{w} : \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p} : \omega_1 = \omega_3, \omega_2 \neq \omega_4$
- $\dot{q} : \omega_1 \neq \omega_3, \omega_2 = \omega_4$
- $\dot{r} : \omega_1 = \omega_3 \neq \omega_2 = \omega_4$
- $\dot{u} = \dot{q}$



Quadrotor

Achieving flight in plus configuration

- $\dot{w} : \omega_1 = \omega_2 = \omega_3 = \omega_4$
- $\dot{p} : \omega_1 = \omega_3, \omega_2 \neq \omega_4$
- $\dot{q} : \omega_1 \neq \omega_3, \omega_2 = \omega_4$
- $\dot{r} : \omega_1 = \omega_3 \neq \omega_2 = \omega_4$
- $\dot{u} = \dot{q}$
- $\dot{v} = \dot{p}$



Quadrotor

Produced forces

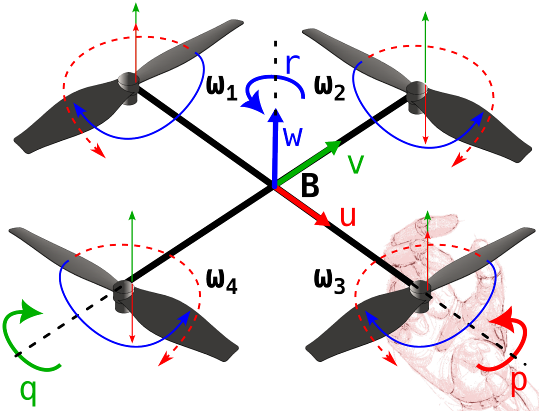
The forces produced are calculated as follows:

$$U_w = F_1 + F_2 + F_3 + F_4$$

$$U_p = l(F_4 - F_2)$$

$$U_q = l(F_1 - F_3)$$

$$U_r = \tau_1 + \tau_3 - \tau_2 - \tau_4$$



Quadrotor

Produced forces

The forces produced are calculated as follows:

$$U_w = F_1 + F_2 + F_3 + F_4$$

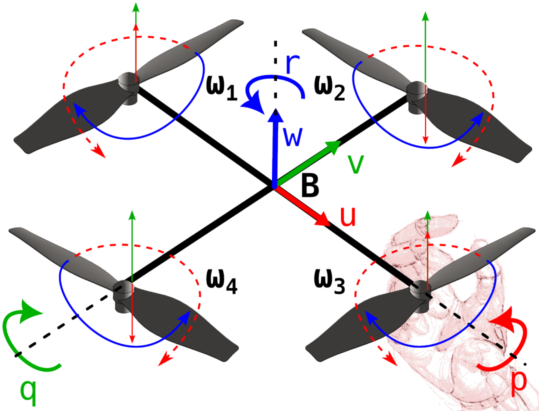
$$U_p = l(F_4 - F_2)$$

$$U_q = l(F_1 - F_3)$$

$$U_r = \tau_1 + \tau_3 - \tau_2 - \tau_4$$

Remember:

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



Quadrotor

Produced forces

The forces produced are calculated as follows:

$$U_w = F_1 + F_2 + F_3 + F_4$$

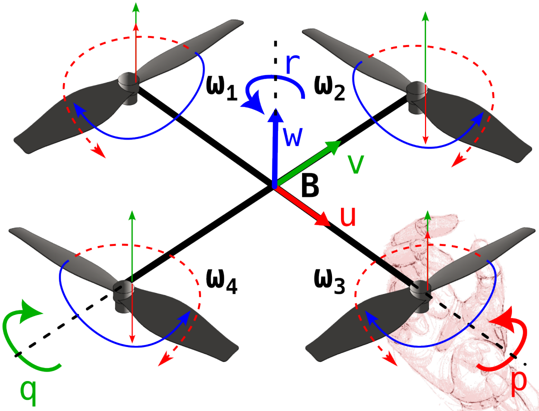
$$U_p = l(F_4 - F_2)$$

$$U_q = l(F_1 - F_3)$$

$$U_r = \tau_1 + \tau_3 - \tau_2 - \tau_4$$

Remember:

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

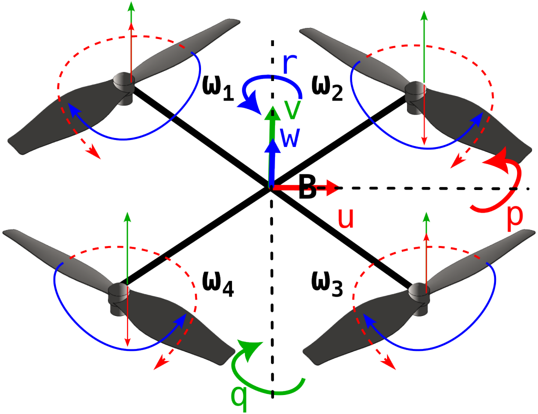


$$U_E = R_E^B U_B$$

Quadrotor

Produced forces

For the cross configurations



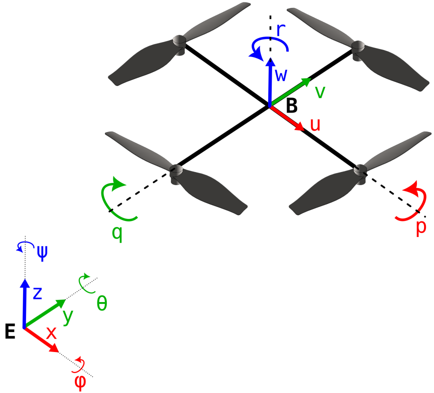
$$U_w = F_1 + F_2 + F_3 + F_4$$

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

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

$$U_r = \tau_1 + \tau_3 - \tau_2 - \tau_4$$

Remember:



- x, y, z: translation along principal axes of the earth frame
- ϕ, θ, ψ: rotation around principal axes of the earth frame
- u, v, w: velocity along principal axes of the drone body frame
- p, q, r: angular velocity around principal axes of the drone body frame

$P = [\xi, \eta]^T$, where: $\xi = [x, y, z]^T$ and $\eta = [\phi, \theta, \psi]^T$
$V = [\pi, \rho]^T$, where: $\pi = [u, v, p]^T$ and $\rho = [p, q, r]^T$

Dynamic modeling

The Lagrangian

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

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

Dynamic modeling

The Lagrangian

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

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

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

Dynamic modeling

The Lagrangian

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

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

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

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{P}} - \frac{\partial L}{\partial P} = \begin{bmatrix} U_{lin} \\ U_{ang} \end{bmatrix}$$

Dynamic modeling

The Lagrangian

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

$$\frac{d}{dt}\frac{\partial\left(T_{lin}+P\right)}{\partial\dot{\xi}}-\frac{\partial\left(T_{lin}+P\right)}{\partial\xi}=U_{lin}$$

$$\frac{d}{dt}\frac{\partial T_{ang}}{\partial\dot{\eta}}-\frac{\partial T_{ang}}{\partial\eta}=U_{ang}$$

Dynamic modeling

The Lagrangian

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

The first equation is easy to differenciate:

Dynamic modeling

The Lagrangian

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

The first equation is easy to differenciate:

$$U_{lin} = m\ddot{\xi} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

Dynamic modeling

The Lagrangian

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

The first equation is easy to differenciate:

$$U_{lin} = m\ddot{\xi} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

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

Dynamic modeling

The Lagrangian

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

The first equation is easy to differenciate:

$$U_{lin} = m\ddot{\xi} + \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}$$

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

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

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:

$$U_{lin} = \begin{bmatrix} 0 \\ 0 \\ U_w \end{bmatrix}$$

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:

$$U_{lin} = \begin{bmatrix} 0 \\ 0 \\ U_w \end{bmatrix}$$

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

$$U_{lin} = R_E^B \begin{bmatrix} 0 \\ 0 \\ U_w \end{bmatrix}$$

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

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

$$\ddot{x} = \left[s_\phi s_\psi + c_\phi c_\psi s_\theta \right] \frac{U_w}{m}$$
$$\ddot{y} = \left[c_\phi s_\theta s_\psi - c_\psi s_\phi \right] \frac{U_w}{m}$$
$$\ddot{z} = -g + c_\phi c_\theta \frac{U_w}{m}$$

Dynamic modeling

The torques

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

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

Dynamic modeling

The torques

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

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

Dynamic modeling

The torques

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

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

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

Dynamic modeling

The torques

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

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

$$\begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = I^{-1}(\eta)\left(\begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix} - C(\eta, \dot{\eta})\dot{\eta} \right) \text{ Where: } \begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix} = R_E^B \begin{bmatrix} U_p \\ U_q \\ U_r \end{bmatrix}$$

Dynamic modeling

Putting it all together

And all together:

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

Dynamic modeling

Simplifications

Since we consider the hovering motion, this means that the ϕ and θ 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 ϕ and θ 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 ϕ and θ 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 $U_z = mg + \Delta U_z$ Therefore, our equations of motion become:

$$\ddot{x} = c_\psi \theta g$$
$$\ddot{y} = -s_\psi \phi g$$
$$\ddot{z} = \frac{\Delta U_z}{m}$$

$$\ddot{\phi} = \frac{c_\psi}{I_{xx}} U_p$$
$$\ddot{\theta} = \frac{s_\psi}{I_{yy}} U_q$$
$$\ddot{\psi} = \frac{1}{I_{zz}} U_r$$

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

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

Tassos Natsakis tassos.natsakis@aut.utcluj.ro

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)

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

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

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

$$s = \left[\xi, \dot{\xi}\right]^T$$

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

The model can be writen in state space as:

$$\dot{s} = As + Bu$$

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:

$$u = -Ks$$

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:

$$u = -Ks$$

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

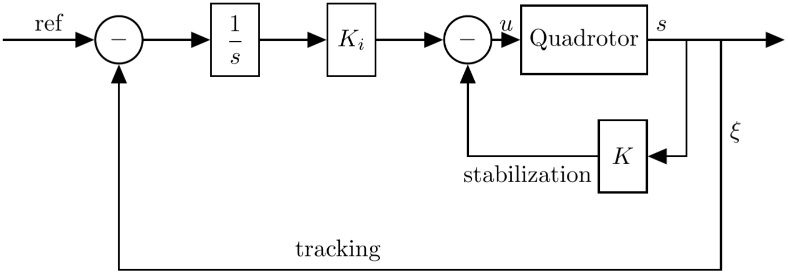
$$u = -Ks + u_o$$
$$u_o = \left[u_z, u_y, u_x, 0\right]^T$$

where:

$$u_x = k_{ix} \int_0^t (r_x - x)\, dt, u_y = k_{iy} \int_0^t (r_y - y)\, dt, u_z = ki_z \int_0^t (r_z - z)\, dt$$

Quadrotor

State feedback

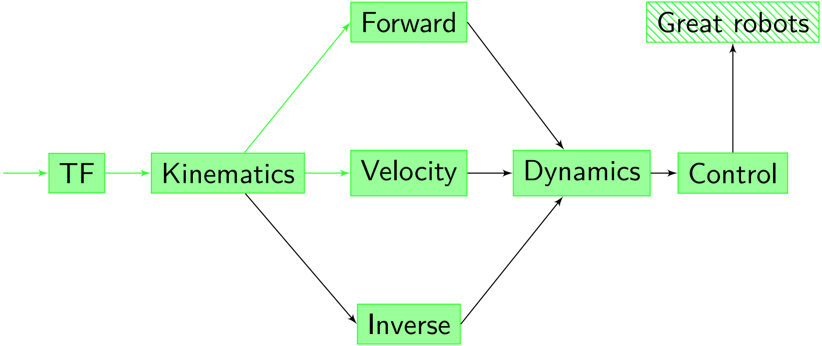


Further reading/watching

Very cool TED talk on drone modeling:
https://www.youtube.com/watch?v=w2itwFJCgFQ

Grand scheme

The big picture



Questions?