Modelado Matemático y Simulación de un UAV Quadrotor como Sistema Dinámico No Lineal con Control en Cascada

Carlos Vidal Rodríguez [1] **Guillermo García Blázquez** [1] **Santiago Álvarez Geanta** [1]

Abstract

Nuestro trabajo presenta el modelado y simulación de un UAV quadrotor como un sistema dinámico no lineal de doce estados. La implementación, realizada en Python, utiliza una arquitectura de control en cascada (Posición-Actitud). En cuanto a la validación numérica, hemos utilizado el integrador *Runge-Kutta (RK45)* de la libería `scikit-learn`, junto a un seguimiento de trayectorias por waypoints y robustez ante efectos aerodinámicos de arrastre.

1.Introducción

La simulación de sistemas dinámicos es una herramienta fundamental en la ingeniería moderna, permitiendo validar comportamientos físicos complejos antes de la implementación en hardware real. En el contexto de la asignatura *Modelos Computacionales y Simulación de Sistemas*, hemos propuesto el estudio de un quadrotor: un sistema subactuado (4 motores para 6 grados de libertad) no lineal.

La motivación principal de nuestro proyecto reside en la creación de un sistema dinámico universal y escalable. Inspirados por la complejidad de la navegación autónoma y competiciones de drones, el objetivo es lograr que el agente recorra un circuito de *waypoints* predefinido de manera autónoma.

2.Estado del Arte

El modelado de quadrotors ha sido extensamente estudiado en la literatura de robótica. Los enfoques clásicos utilizan el formalismo de Euler-Lagrange o Newton-Euler para derivar las ecuaciones de movimiento (Luukkonen, 2011). Mientras que los modelos lineales simplificados son útiles para el vuelo estacionario (*hovering*), la simulación de maniobras agresivas requiere modelos no lineales completos que consideren las matrices de rotación $SO(3)$, tal como se implementa en este trabajo.

En cuanto al control, la arquitectura en cascada es el estándar industrial. Autores como Mahony et al. (Mahony et al., 2012) establecieron la separación de escalas temporales: un bucle interno rápido para la actitud y un bucle externo más lento para la posición. Aunque existen técnicas avanzadas como el Control Predictivo de Modelos (MPC), el control PID sigue siendo predominante por su robustez (Hoffmann et al., 2007), siendo suficiente para validar el modelo dinámico propuesto.

3.Modelado Matemático del Sistema

El sistema se modela como un cuerpo rígido con masa m y una matriz de inercia I. El vector de estado $\mathbf{x} \in \mathbb{R}^{12}$ se define como:

$$\mathbf{x}(t) = [x, y, z, v_x, v_y, v_z, \phi, \theta, \psi, p, q, r]^T \qquad (1)$$

Donde:

- $\xi = [x, y, z]^T$: Posición en el marco inercial (Mundo).
- $v = [v_x, v_y, v_z]^T$: Velocidad lineal en el marco inercial.
- $\eta = [\phi, \theta, \psi]^T$: Ángulos de Euler (Roll, Pitch, Yaw; Figura 1).
- $\Omega = [p, q, r]^T$: Velocidad angular en el marco del cuerpo.

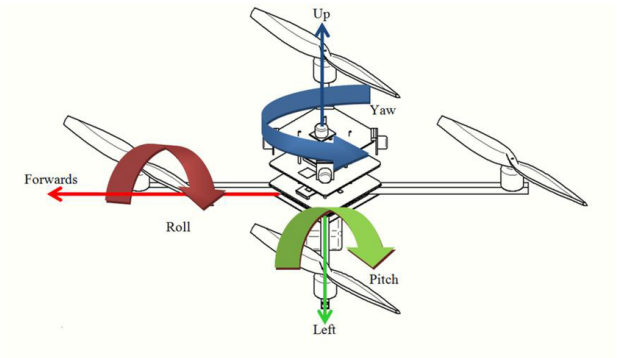

Figure 1: Esquema del dron y definición de los ejes y ángulos de orientación (roll, pitch y yaw).

3.1.Cinemática y Matrices de Rotación

Para transformar vectores del marco del cuerpo (B) al marco del mundo (W), utilizamos la matriz de rotación R_{BW} (YouTube, 2020), definida por la secuencia de rotaciones $Z \to Y \to X$ (Yaw-Pitch-Roll):

$$R_{BW} = \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\psi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - c\psi s\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{2}$$

donde c y s representan $\cos()$ y $\sin()$.

La relación cinemática entre la tasa de cambio de los ángulos de Euler ($\dot{\eta}$) y la velocidad angular del cuerpo (Ω) no es directa, sino que está dada por la matriz de transformación:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \frac{\sin\phi}{\cos\theta} & \frac{\cos\phi}{\cos\theta} \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3}$$

Es importante notar que este mapeo presenta una singularidad (*Gimbal Lock*) cuando $\theta = \pm\pi/2$, una limitación conocida de la representación de Euler que nuestro controlador evita limitando los ángulos de operación.

3.2.Dinámica Traslacional

La evolución de la velocidad lineal se rige por la segunda ley de Newton, considerando la gravedad, el empuje total T generado por los rotores y una fuerza de arrastre aerodinámico lineal (k_{drag}):

$$\dot{v} = g \cdot \mathbf{e}_z + \frac{1}{m} R_{BW} \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix} - k_{drag} \cdot v \tag{4}$$

Donde $\mathbf{e}_z = [0, 0, -1]^T$ es el vector de gravedad en el marco inercial. Es crucial destacar que el empuje T solo actúa en el eje Z local del dron, para moverse lateralmente, el dron debe rotar R_{BW}, proyectando parte de ese vector empuje sobre los planos X e Y del mundo.

3.3.Dinámica Rotacional (Leyes de Euler)

La dinámica angular es la parte más compleja del sistema debido a los efectos de acoplamiento. Basándonos en las ecuaciones de Euler para cuerpo rígido:

$$I \cdot \dot{\Omega} = -\Omega \times (I \cdot \Omega) + \tau \tag{5}$$

Desarrollando esta ecuación vectorial para nuestros ejes principales, obtenemos las ecuaciones implementadas en el código (dynamics.py), donde se aprecian explícitamente los términos giroscópicos:

$$\dot{p} = \frac{I_{yy} - I_{zz}}{I_{xx}} qr + \frac{\tau_\phi}{I_{xx}} \tag{6}$$

$$\dot{q} = \frac{I_{zz} - I_{xx}}{I_{yy}} pr + \frac{\tau_\theta}{I_{yy}} \tag{7}$$

$$\dot{r} = \frac{I_{xx} - I_{yy}}{I_{zz}} pq + \frac{\tau_\psi}{I_{zz}} \tag{8}$$

Estos términos cruzados (por ejemplo, $\frac{I_{yy}-I_{zz}}{I_{xx}} qr$) implican que una rotación simultánea en Pitch (q) y Yaw (r) inducirá una aceleración no deseada en Roll (p). Nuestro controlador debe ser lo suficientemente robusto para compensar estos fenómenos naturales del sistema dinámico.

3.4.Arquitectura de Control

El control se divide en dos bucles anidados.

1. Control de Posición (Bucle Externo): Dado un error de posición $e_p = p_{des} - p$, calculamos una fuerza virtual deseada mediante un control PD:

$$F_{des} = K_{P_{pos}} e_p + K_{D_{pos}} (v_{des} - v) \tag{9}$$

Esta fuerza se traduce en inclinaciones deseadas (ϕ_{des}, θ_{des}). Por ejemplo, para acelerar en el eje X positivo, el dron debe realizar un Pitch negativo (bajar el morro).

2. Control de Actitud (Bucle Interno): Dado el error de ángulo, calculamos los torques físicos τ necesarios:

$$\tau = K_{P_{att}} (\eta_{des} - \eta) + K_{D_{att}} (\Omega_{des} - \Omega) \tag{10}$$

4.Implementación y Simulación

La simulación la hemos implementado en Python utilizando la librería `scipy.integrate`.

4.1.Parámetros del Sistema

Para asegurar realismo, los valores de inercia y masa se han derivado de un modelo físico de referencia escalable según la longitud del brazo L.

Table 1: Parámetros Físicos del Dron Simulado

Parámetro	Símbolo	Valor
Masa Total	m	0,32 kg
Longitud brazo	L	0,20 m
Inercia X	I_{xx}	0,0032 kg·m^2
Inercia Y	I_{yy}	0,0032 kg·m^2
Inercia Z	I_{zz}	0,0064 kg·m^2
Coef. Arrastre	k_{drag}	0,75
Gravedad	g	9,81 m/s^2

4.2.Algoritmo de Simulación

El bucle principal gestiona la lógica discreta de los waypoints y la integración continua de la física. Se utiliza el integrador `RK45` con una evaluación en pasos fijos de $dt = 0{,}01s$ para la recolección de datos.

5.Resultados y Análisis de Telemetría

Para validar el modelo, hemos diseñado un circuito de prueba con cambios de altura y desplazamientos laterales. La Figura **??** muestra la reconstrucción espacial del vuelo. Observamos que el dron completa el circuito pasando por los puntos $(0, 0, 10)$, $(5, 0, 5)$, $(0, 5, 0)$ y volviendo al origen.

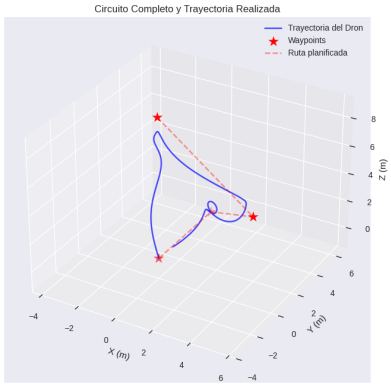

Figure 2: Trayectoria realizada por el dron.

El análisis detallado de la dinámica se realiza sobre la Figura 3, que desglosa el comportamiento temporal:

Análisis de Posición (Filas 1-3): Las gráficas superiores muestran la evolución de X, Y, Z, y podemos observa un comportamiento suave y amortiguado.

- En $t \approx 6s$ (primera línea verde), el dron alcanza el primer waypoint (subida vertical a $Z = 10$). Podemos ver que X e Y se mantienen en 0, lo cual es correcto.
- Entre $t = 6s$ y $t = 10s$, el objetivo cambia a $X = 5$. Vemos como la curva azul de X crece hasta converger en 5.
- Se aprecia un ligero sobreimpulso (*overshoot*) en las curvas de posición antes de estabilizarse, característico de los controladores PD sintonizados para una respuesta rápida (K_p alto).

Análisis de Actitud (Filas 4-6): Aquí reside la validación del sistema dinámico acoplado.

- **Correlación Pitch-X:** Observando el intervalo $t \in [6, 10]$, donde el dron avanza en X, la gráfica de *Pitch* (θ) muestra una deflexión negativa significativa. Esto valida el modelo físico: para avanzar hacia adelante ($+X$), el dron debe inclinar el morro hacia abajo.
- **Correlación Roll-Y:** De forma análoga, cuando el dron se mueve en el eje Y (alrededor de $t = 12s$), se observa una gran oscilación en la gráfica de *Roll* (ϕ).
- **Estabilidad de Yaw:** A pesar de las perturbaciones generadas por los términos de acoplamiento $p \cdot q$ descritos en la Ec. (8), el ángulo de *Yaw* (ψ) se mantiene cercano a 0, demostrando que el término de control τ_ψ compensa eficazmente los torques inducidos.

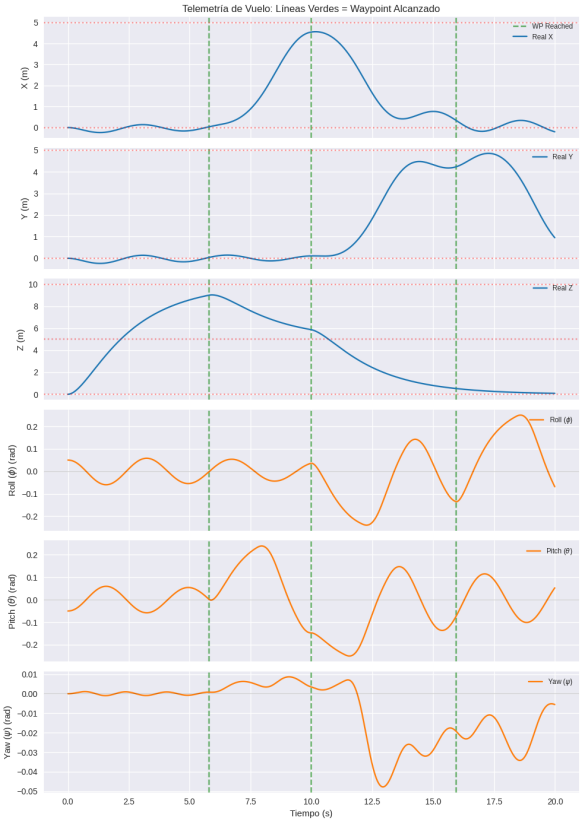

Figure 3: Telemetría de vuelo. Filas 1-3: Posición Real vs Waypoints. Filas 4-6: Evolución de los Ángulos de Euler. Las líneas verticales verdes indican el instante en que se alcanza un waypoint.

Finalmente, la convergencia de los errores a cero tras alcanzar cada waypoint confirma que el integrador numérico maneja correctamente la rigidez de las ecuaciones y que el control en cascada es estable.

6.Conclusiones

En este trabajo se ha modelado, implementado y validado un sistema dinámico complejo correspondiente a un UAV quadrotor. Hemos demostrado que la formulación mediante ecuaciones de Newton-Euler captura correctamente la física del vuelo, incluyendo la necesidad de modificar la actitud para generar aceleraciones traslacionales. El control en cascada permite desacoplar efectivamente la dinámica rápida (rotacional) de la lenta (traslacional), simplificando la tarea de navegación. La simulación numérica es una herramienta viable para el ajuste de ganancias de control sin riesgo de daño material.

Referencias

Hoffmann, G. M., Huang, H., Waslander, S. L., and Tomlin, C. J. Quadrotor helicopter flight dynamics and control: Theory and experiment. In *AIAA Guidance, Navigation and Control Conference*, 2007.

Luukkonen, T. Modelling and control of quadcopter. *Aalto University, Independent Research Project*, 2011.

Mahony, R., Kumar, V., and Corke, P. Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor. *IEEE Robotics & Automation Magazine*, 19(3):20–32, 2012.

YouTube. 3d rotation matrices of x, y, z. `https://www.youtube.com/watch?v=uLl_egj9F2M`, 2020. Video en YouTube.