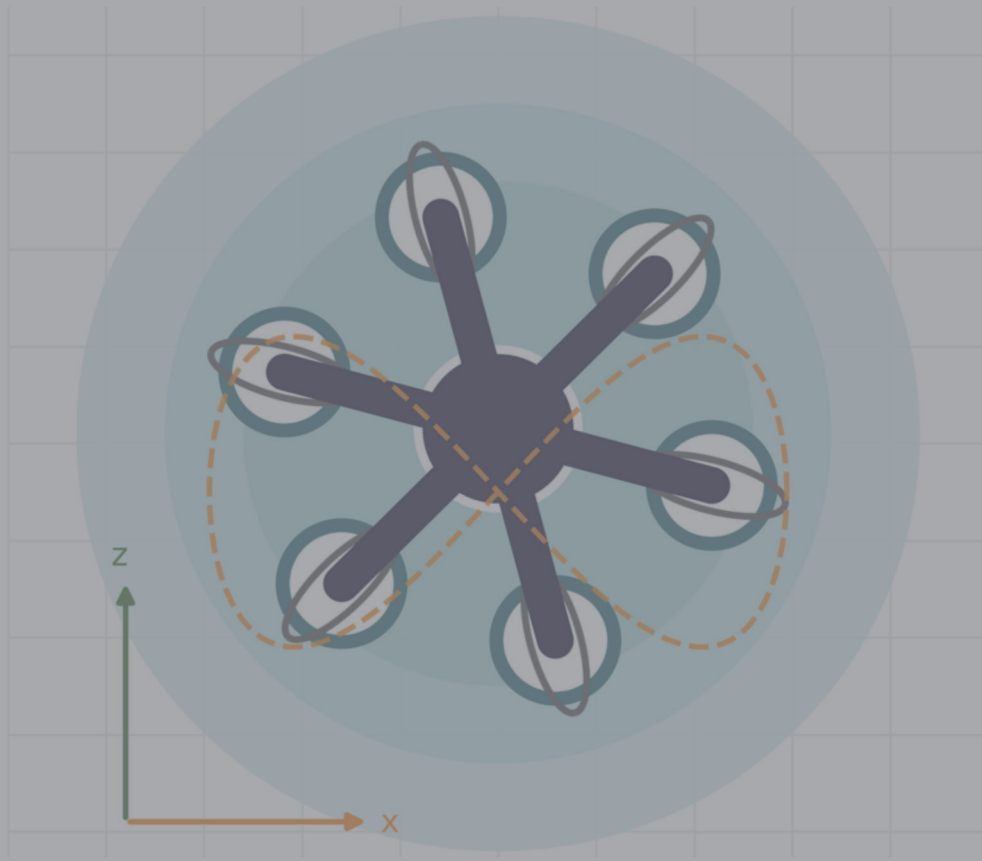
MODELACIÓN MATEMÁTICA Y CONTROL DE ROBOTS AÉREOS MULTIRROTOR



JOSÉ VARELA-ALDÁS

Universidad Indoamérica


Z
X

Modelación matemática y control de robots aéreos multirrotor

Fecha de publicación: 23 de octubre de 2026

Autoridades

Ing. Saúl Lara Paredes, PhD – Canciller
Ing. Janio Jadán Guerrero, PhD – Rector
Dr. Daniel Simancas, PhD – Vicerrector de Investigación
Ing. Ignacio Ayala, PhD – Vicerrector Académico y de Vinculación
Ing. Aidé Naranjo, Mgs. – Vicerrectora Administrativa y de Aseguramiento de la Calidad

© Autores: Jose Varela-Aldás[1]

[1]Centro de Investigación en Mecatrónica y Sistemas Interactivos (MIST), Universidad Tecnológica Indoamérica, Ambato, Ecuador.
josevarela@uti.edu.ec

ISBN: 978-9942-693-24-2
Registro SENADI: N° QUI-073686

Revisado y aprobado para su publicación por el Comité Editorial de la Universidad Tecnológica Indoamérica (Quito, Ecuador) y por los revisores Francisco Yumbla, PhD (Escuela Superior Politécnica del Litoral) y David Rivas, PhD (Universidad de las Fuerzas Armadas).

Editor: Ing. Hugo Arias Flores, MBA

Editorial de la Universidad Tecnológica Indoamérica. Quito, Ecuador.



Aviso Legal

Queda rigurosamente prohibida la reproducción total o parcial de esta obra por cualquier medio o procedimiento, comprendidos la fotocopia y el tratamiento informático, sin autorización escrita del titular del copyright, bajo las sanciones previstas por las leyes. El contenido de este libro incluyendo textos, imágenes, gráficos, tablas, cuadros y referencias bibliográficas es de exclusiva responsabilidad del/ de los autor (es). Las opiniones, datos y criterios expresados no representan necesariamente la postura institucional ni el pensamiento de la Editorial Universidad Tecnológica Indoamérica.

Para citar este libro:

Varela-Aldás, J. (2026). *Modelación matemática y control de robots aéreos multirrotor.* Editorial Universidad Tecnológica Indoamérica.

Modelación matemática y control de robots aéreos multirrotor

José Varela-Aldás

Índice de contenidos

Introducción general 15
Capítulo I. Fundamentos de la robótica aérea 17
1. Robots aéreos multirrotor y el problema de control 19
2. Fundamentos matemáticos y representaciones espaciales 25
3. Modelación cinemática del multirrotor 33
Capítulo II. Modelación e identificación 39
4. Modelación dinámica y modelos reducidos 41
5. Identificación de sistemas multirrotor 48
6. Identificación en línea y control adaptativo 56
Capítulo III. Control de vuelo y seguridad 63
7. Controladores de realimentación 65
8. Control predictivo no lineal 73
9. Dinámica no modelada, observadores y adaptación 83
10. Seguimiento seguro de caminos 91
Capítulo IV. Percepción, implementación y diseño integrado 101
11. Servo visual predictivo 103
12. Implementación y validación experimental 111
13. Proceso integrado y decisiones de diseño 119
14. Guía de reproducción y transferencia experimental 126
Conclusiones 163
Glosario 165

Referencias 169
Anexo A. Notación matemática 173
Anexo B. Parámetros y resultados reportados 175

Índice de figuras

Figura 1. Mapa conceptual del libro: la autonomía aérea surge de la integración de varias capacidades 16

Figura 2. Configuraciones en X de un cuadricóptero y un hexacóptero 21

Figura 3. Capas funcionales de una arquitectura de autonomía aérea 23

Figura 4. Marcos de referencia inercial y solidario al vehículo 26

Figura 5. Comparación conceptual entre ángulos de Euler y cuaterniones 29

Figura 6. Efecto conceptual del método de integración sobre la precisión y la estabilidad numérica 31

Figura 7. Flujo de la modelación cinemática: de velocidades del cuerpo a evolución de la pose 34

Figura 8. Diferencia entre camino y trayectoria 36

Figura 9. Fuerzas y momentos producidos por diferencias de velocidad entre rotores 42

Figura 10. Tres niveles complementarios de representación: físico, reducido y basado en datos 44

Figura 11. Modelación: proceso iterativo que alterna hipótesis, datos, validación y refinamiento 50

Figura 12. Proceso de identificación DMDc desde datos de vuelo hasta un modelo lineal controlado 52

Figura 13. Error integral cuadrático reportado para los modelos Euler-Lagrange y DMD del hexacóptero 53

Figura 14. Ventana móvil para actualizar un modelo DMDc con las mediciones más recientes 58

Figura 15. Lazo básico de control por realimentación aplicado a un multirrotor 66

Figura 16. Compromiso cualitativo entre precisión, consideración de restricciones y costo computacional 69

Figura 17. Tiempos promedio de cálculo reportados para los tres controladores..... 70

Figura 18. Horizonte deslizante: en cada instante se optimiza una secuencia y se aplica únicamente la primera acción 74

Figura 19. Restricción geométrica de seguridad alrededor de un obstáculo..... 78

Figura 20. Arquitectura general del NMPC: medición, predicción, optimización y aplicación de la primera acción 79

Figura 21. Arquitectura del NMPC adaptativo con observador dinámico de momento 86

Figura 22. Reconstrucción conceptual de la perturbación mediante modelo nominal, observador y adaptación..... 87

Figura 23. Descomposición del error en componente de contorno y componente de avance 91

Figura 24. El mapa Log convierte el error de cuaternión en un vector del espacio tangente 93

Figura 25. La CLF impulsa el estado hacia el objetivo y la CBF impide abandonar el conjunto seguro 95

Figura 26. Arquitectura integrada del NMPCC con errores geométricos, CLF, CBF y control de actitud.....96

Figura 27. Proyección perspectiva de un punto tridimensional sobre el plano de la imagen 105

Figura 28. Comparación conceptual entre un controlador que no impone visibilidad y un MPC que mantiene margen en el campo de visión 107

Figura 29. Arquitectura del servo visual predictivo: imagen, detección de rasgos, modelo aumentado, optimización y UAV 108

Figura 30. Escalera de validación desde el análisis matemático hasta el vuelo real 112

Figura 31. Pila de software y hardware para simulación, HiL y vuelo real 114

Figura 32. Flujo práctico desde los requisitos hasta la validación y la mejora iterativa 120

Índice de tablas

Tabla 1. Configuraciones multirrotor y características generales 21

Tabla 2. Dificultades del control y sus consecuencias prácticas 22

Tabla 3. Notación básica adoptada en el libro 26

Tabla 4. Representaciones de orientación 29

Tabla 5. Niveles de modelación y uso recomendado 45

Tabla 6. Comparación de métodos de identificación empleados en los casos de estudio 52

Tabla 7. Error ISE de los modelos identificados en el hexacóptero 54

Tabla 8. Comparación de controladores de realimentación y control predictivo 69

Tabla 9. Tiempos promedio de cálculo reportados 70

Tabla 10. Restricciones declaradas en los controladores predictivos analizados 77

Tabla 11. Estrategias frente a dinámica no modelada 84

Tabla 12. Diferencia entre función de Lyapunov y función de barrera 95

Tabla 13. Modalidades de servo visual 104

Tabla 14. Niveles de validación 112

Tabla 15. Selección de controlador por escenario 121

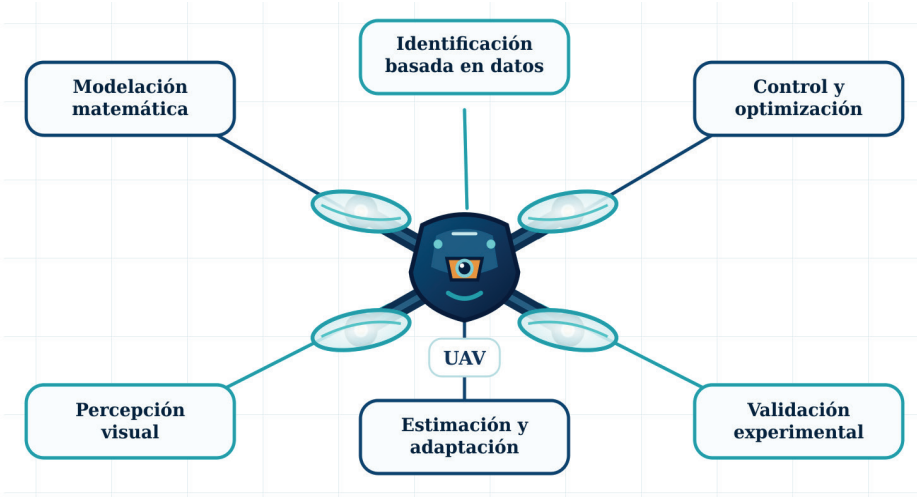
Introducción general

La modelación matemática y el control de un robot aéreo multirrotor forman una cadena inseparable. Un controlador necesita predecir, aunque sea de manera aproximada, cómo responderá el vehículo. El modelo, a su vez, solo es útil si representa las variables que el sistema puede medir y las órdenes que realmente puede ejecutar. En plataformas comerciales, por ejemplo, el investigador rara vez controla directamente los torques de cada rotor. Lo habitual es enviar referencias de velocidad o posición a un controlador interno. Esa realidad modifica el tipo de modelo que conviene desarrollar. El propósito no es presentar recetas aisladas, sino explicar qué información debe conservarse entre la misión, el modelo, el optimizador y la validación para que una decisión técnica pueda revisarse y reproducirse.

La Figura 1 presenta el mapa conceptual de la obra. La autonomía no aparece por añadir un algoritmo aislado, sino por integrar modelación, identificación, control, percepción y validación.

Figura 1.

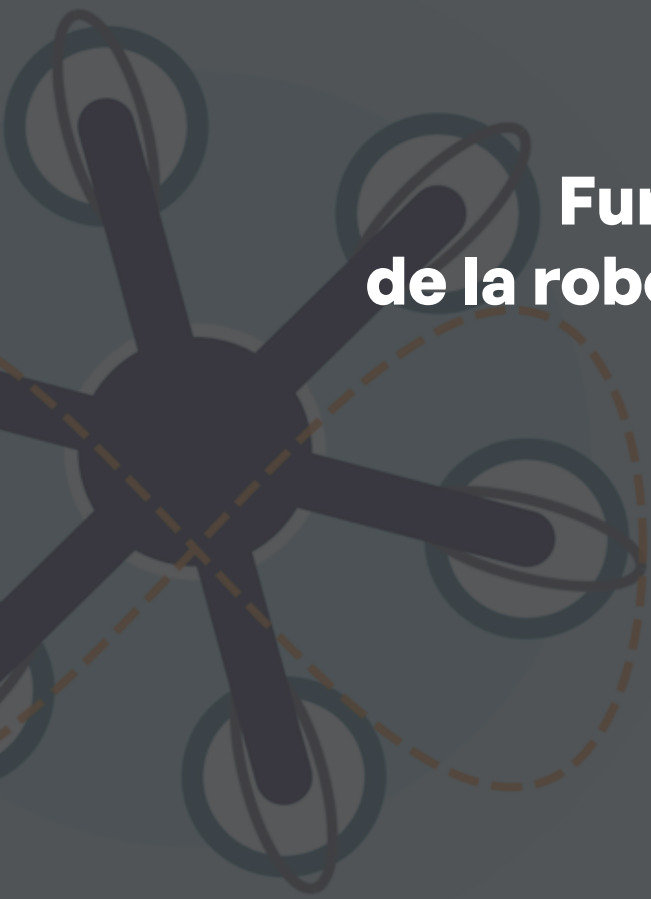
Mapa conceptual del libro: la autonomía aérea surge de la integración de varias capacidades



El lector encontrará dos ideas transversales. La primera es que no existe un único modelo correcto. Un modelo físico completo sirve para analizar fuerzas y energía, mientras que uno reducido puede ser mejor para implementar un controlador de alto nivel. Un modelo basado en datos puede predecir con gran precisión alrededor del régimen observado, pero requiere datos suficientemente informativos. La segunda idea es que la complejidad del controlador debe justificarse por la tarea. Por ejemplo, un regulador cinemático puede ser excelente para vuelos suaves y con pocas restricciones; un control predictivo no lineal (NMPC) resulta conveniente cuando la trayectoria es agresiva, existen límites duros o deben considerarse obstáculos.

Capítulo I

Fundamentos de la robótica aérea




Z
X

1. Robots aéreos multirrotor y el problema de control

Objetivos del capítulo. Reconocer las configuraciones multirrotor, comprender por qué son sistemas subactuados y no lineales, distinguir las capas de una arquitectura de vuelo y relacionar la dificultad de la misión con la elección del modelo y del controlador. La selección final debe partir de la misión, del espacio disponible y de los sensores, no solo del número de rotores.

1.1 El robot aéreo multirrotor

Un multirrotor es un vehículo aéreo de despegue y aterrizaje vertical que se produce mediante varias hélices. A diferencia de un avión convencional, no necesita una velocidad horizontal mínima para mantenerse en el aire. Puede despegar verticalmente, detenerse en un punto, desplazarse lateralmente y aterrizar en espacios reducidos. Estas propiedades explican su uso en inspección, agricultura, búsqueda, vigilancia, cartografía, transporte ligero y experimentación robótica.

Las configuraciones más habituales se nombran por el número de rotores. El cuadricóptero utiliza cuatro; el hexacóptero, seis; y el octocóptero, ocho. Aumentar el número de rotores permite elevar la

carga útil y añadir redundancia, pero incrementa masa, consumo, costo y complejidad. Los trabajos de esta obra utilizan principalmente la familia Da-Jiang Innovations (DJI) Matrice: un Matrice 100 de cuatro rotores y un 600 de seis rotores. La Figura 2 ilustra las configuraciones en X de cuatro y seis rotores, mientras que la Tabla 1 resume sus características generales.

1.2 Control de un multirrotor

Un multirrotor se mueve en un espacio de seis grados de libertad: tres posiciones y tres orientaciones. Sin embargo, en una formulación clásica dispone de cuatro entradas independientes: empuje total y tres momentos. Por esta razón, se lo describe como un sistema **subactuado**. Para avanzar en el eje horizontal, el vehículo debe inclinarse; la posición y la orientación están acopladas.

La dinámica también es no lineal. Las rotaciones cambian la dirección del empuje respecto al marco inercial; aparecen productos entre velocidades angulares y lineales y los efectos aerodinámicos aumentan con la velocidad. A esto se añaden saturaciones de motores, retardos de comunicación, ruido de sensores, viento, variación de carga, desgaste y simplificaciones inevitables del modelo. La Tabla 2 sintetiza estas dificultades y las respuestas de diseño asociadas. Las funciones de barrera de control (Control Barrier Functions, CBF) permiten formular condiciones explícitas de seguridad.

Figura 2.

Configuraciones en X de un cuadricóptero y un hexacóptero

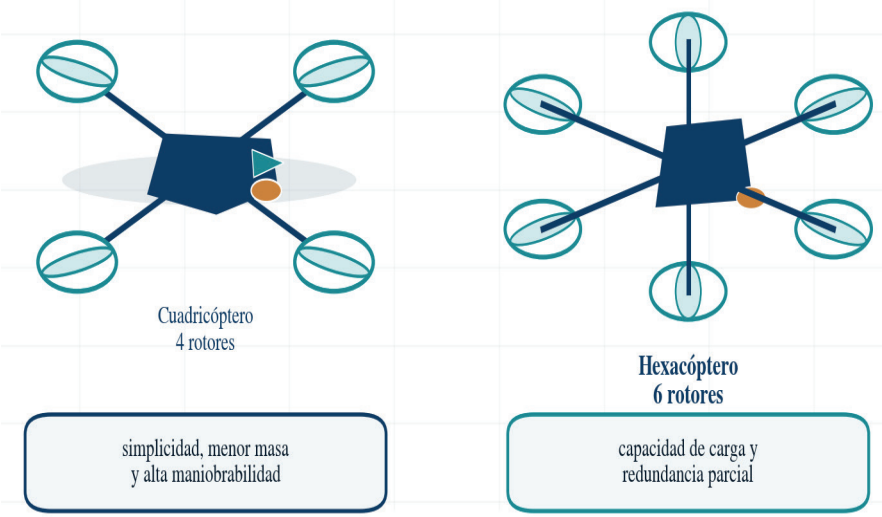


Tabla 1. *Configuraciones multirrotor y características generales*

Configuración	**Rotores**	**Ventajas principales**	**Limitaciones frecuentes**
Cuadricóptero	4	Bajo peso, costo moderado, gran disponibilidad	Menor redundancia y carga útil
Hexacóptero	6	Mayor empuje, mejor capacidad de transporte, tolerancia parcial a fallos	Mayor masa, consumo y complejidad dinámica
Octocóptero	8	Alta redundancia y capacidad de carga	Costo, tamaño y demanda energética elevados

Tabla 2. *Dificultades del control y sus consecuencias prácticas*

Dificultad	Consecuencia en el vuelo	Respuesta de diseño
Subactuación	No se puede ordenar una traslación arbitraria sin cambiar la actitud	Control en cascada y referencias compatibles
No linealidad	Un modelo lineal pierde precisión lejos del vuelo estacionario	Modelos no lineales o identificación alrededor del régimen de operación
Saturación	Las referencias pueden exigir velocidades o fuerzas imposibles	Restricciones explícitas o funciones de saturación
Perturbaciones	Aparecen errores persistentes y oscilaciones	Observadores, adaptación o modelos actualizados en línea
Obstáculos	Un error pequeño puede producir una colisión	Planificación, CBF y restricciones geométricas
Costo computacional	El controlador puede terminar tarde	Horizontes cortos, discretización eficiente y solvers en tiempo real

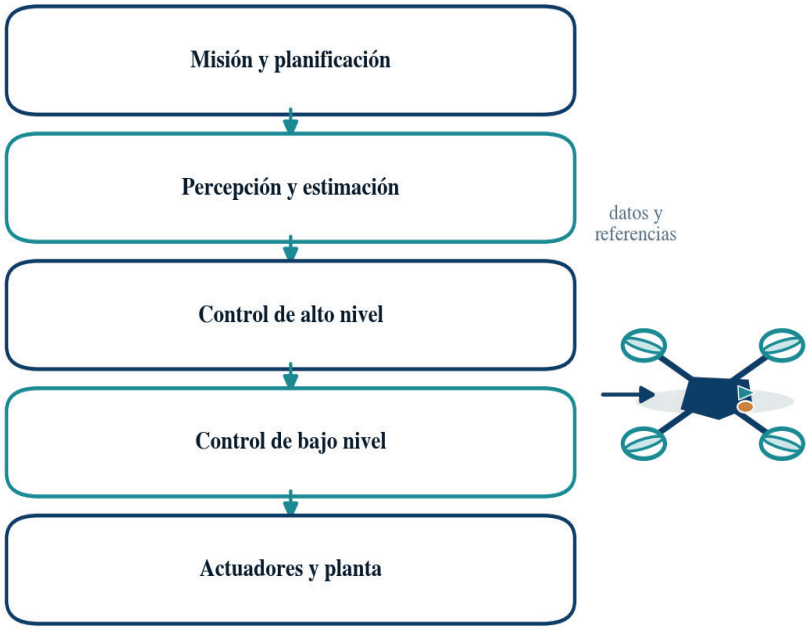
Advertencia. Un controlador que funciona en vuelo estacionario no está automáticamente preparado para una trayectoria rápida. Al aumentar la velocidad, aparecen más acoplamientos, arrastre, retardos y saturaciones.

1.3 Las capas de una arquitectura de vuelo

Una misión autónoma puede entenderse como una pila de capas. En la parte superior se encuentra la misión: inspeccionar una superficie, seguir un camino o mantener un objeto en la imagen. La percepción transforma sensores en información útil; la estimación combina sistema de posicionamiento global (GPS), unidad de medición inercial (IMU), barómetro, cámara u otros sensores; el control de alto nivel

calcula referencias de movimiento; el control de bajo nivel estabiliza actitud y altura; y los actuadores convierten órdenes eléctricas en empuje. La Figura 3 organiza estas funciones en una arquitectura por capas.

Figura 3.
Capas funcionales de una arquitectura de autonomía aérea



En plataformas comerciales, las capas inferiores suelen estar cerradas y ajustadas por el fabricante. El investigador interactúa mediante un kit de desarrollo de software (SDK) y ordena velocidades de maniobrabilidad. El modelo de alto nivel no intenta describir cada pulso de motor, sino la relación entre una velocidad de referencia y la que finalmente alcanza el vehículo.

1.4 Modelar para una tarea concreta

La modelación puede realizarse en tres niveles complementarios. En primer lugar, el **modelo físico completo** explica fuerzas, momentos, energía y acoplamientos. En segundo, el **modelo reducido** agrupa la dinámica interna y utiliza variables accesibles, como velocidades lineales y de guiñada. En tercer lugar, el **modelo basado en datos** aproxima directamente la evolución observada mediante matrices o funciones identificadas.

Ninguno de estos niveles reemplaza por completo a los demás. El modelo físico ayuda a interpretar; el reducido facilita el control; y el modelo de datos corrige discrepancias o se adapta a cambios.

1.5 Resumen del capítulo

Un multirrotor combina gran maniobrabilidad con una dinámica subactuada, no lineal y sensible a perturbaciones. El control debe diseñarse dentro de una arquitectura que ya incluye sensores, estimadores y lazos internos. Por ello, el modelo más útil no siempre es el más detallado: debe corresponder a las variables accesibles y a la tarea que se desea ejecutar.

1.6 Preguntas de repaso

1. ¿Por qué un multirrotor se considera subactuado?
2. ¿Qué ventajas y costos aparecen al pasar de cuatro a seis rotores?
3. ¿Qué diferencia existe entre el control de alto nivel y el de bajo nivel?
4. ¿Cuándo sería preferible un modelo reducido frente a uno completo?

5. Mencione tres perturbaciones capaces de modificar el comportamiento de vuelo.

2. Fundamentos matemáticos y representaciones espaciales

Objetivos del capítulo. Manejar los marcos de referencia, representar posición y orientación, diferenciar ángulos de Euler y cuaterniones, comprender la discretización de ecuaciones diferenciales y establecer una notación común para los capítulos posteriores. Estas convenciones se mantendrán en los modelos, controladores y resultados para que cada ecuación pueda relacionarse con una variable física.

2.1 Vectores, matrices y marcos de referencia

La posición de un vehículo no tiene significado completo sin un marco de referencia. En robótica aérea se utilizan, como mínimo, dos marcos. El marco inercial, denotado por $\langle I \rangle$, se considera fijo al entorno. El marco del cuerpo $\langle B \rangle$ se mueve con el vehículo y suele situar su origen en el centro de masa.

La posición del centro de masa respecto al marco inercial se representa como:

$$\mathbf{p} = [x \quad y \quad z]^T$$

Una velocidad expresada en $\langle B \rangle$ no puede sumarse directamente con otra expresada en $\langle I \rangle$. Antes debe aplicarse una transformación. Este cuidado evita muchos errores de implementación: el sensor puede entregar una medida en un marco, mientras que la trayectoria deseada está definida en otro. La Figura 4 muestra los marcos de referencia utilizados y la Tabla 3 resume la notación básica adoptada.

Figura 4.

Marcos de referencia inercial y solidario al vehículo

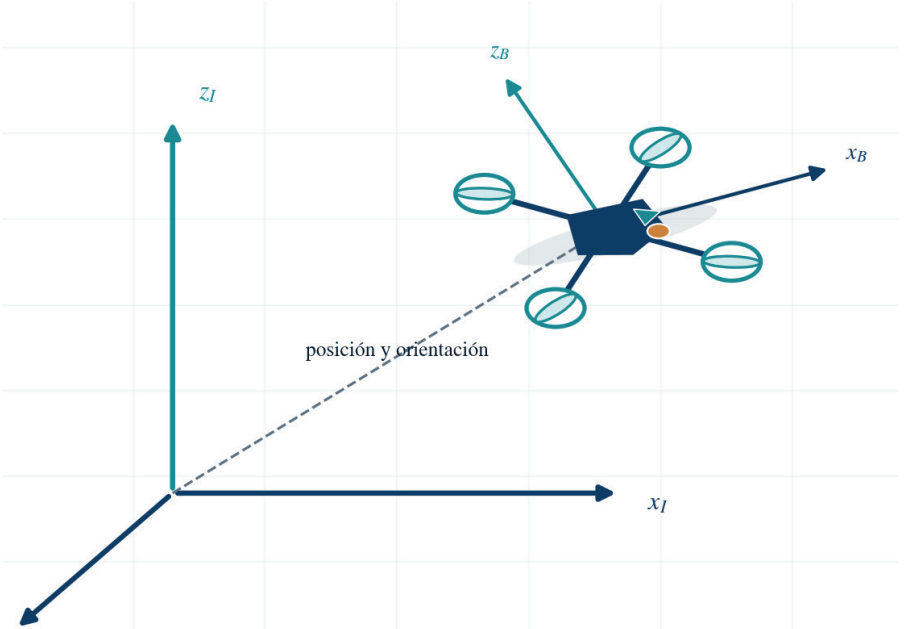


Tabla 3. *Notación básica adoptada en el libro*

Símbolo	Significado	Dimensión típica
p	Posición en el marco inercial	3×1
η	Pose reducida: posición y guiñada	4×1
ν	Velocidades de maniobrabilidad en el cuerpo	4×1
μ	Entrada o referencia de velocidad	4×1
R	Matriz de rotación	3×3
J	Jacobiano cinemático	variable

Símbolo	Significado	Dimensión típica
$\mathbf{M}$	Matriz de masa e inercia reducida	4×4
$\mathbf{C}$	Matriz de Coriolis y términos centrípetos	4×4
$\mathbf{x}$	Estado generalizado	variable

2.2 Matrices de rotación

Una matriz de rotación transforma coordenadas entre marcos ortonormales. Para una rotación de guiñada ψ alrededor del eje vertical puede utilizarse:

$$\mathbf{R}_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

La matriz satisface $\mathbf{R}^T\mathbf{R} = \mathbf{I}$ y $\det(\mathbf{R}) = 1$. Su inversa coincide con su transpuesta. Estas propiedades son útiles tanto para transformar velocidades como para verificar que un cálculo numérico no ha degradado la ortogonalidad.

2.3 Ángulos de Euler

Los ángulos de Euler describen la orientación mediante tres rotaciones consecutivas. En aeronáutica se utilizan *roll* o alabeo ϕ, *pitch* o cabeceo θ, y *yaw* o guiñada ψ. Son intuitivos y fáciles de relacionar con el movimiento, pero dependen del orden de rotación y pueden presentar singularidades. El bloqueo de cardán aparece cuando dos ejes se alinean y se pierde un grado de libertad efectivo.

En varios modelos se conserva únicamente ψ. Esta decisión no significa que el vehículo no cambie *roll* y *pitch*. Implica que los lazos internos mantienen esos ángulos pequeños durante vuelos no

agresivos y que el control de alto nivel los absorbe dentro de la respuesta de velocidad.

2.4 Cuaterniones

Un cuaternión unitario representa orientación mediante cuatro componentes:

$$\mathbf{q} = [q_w \quad q_x \quad q_y \quad q_z]^T, \qquad \| \mathbf{q} \| = 1$$

Su principal ventaja es evitar singularidades y proporcionar transiciones suaves. La dinámica de orientación se expresa como:

$$\dot{\mathbf{q}} = \frac{1}{2} \mathbf{S}(\boldsymbol{\omega})\mathbf{q}$$

En la fórmula, $\boldsymbol{\omega}$ es la velocidad angular y $\mathbf{S}(\boldsymbol{\omega})$ es una matriz construida con sus componentes. En el control de contorno de Guevara *et al.* (2025), el error entre el cuaternión deseado y el real se lleva al espacio tangente mediante el mapa logarítmico. Así, el error de orientación puede penalizarse como un vector tridimensional dentro de una función de costo. La Figura 5 compara ambas representaciones y la Tabla 4 sintetiza sus ventajas y precauciones.

Figura 5.

Comparación conceptual entre ángulos de Euler y cuaterniones

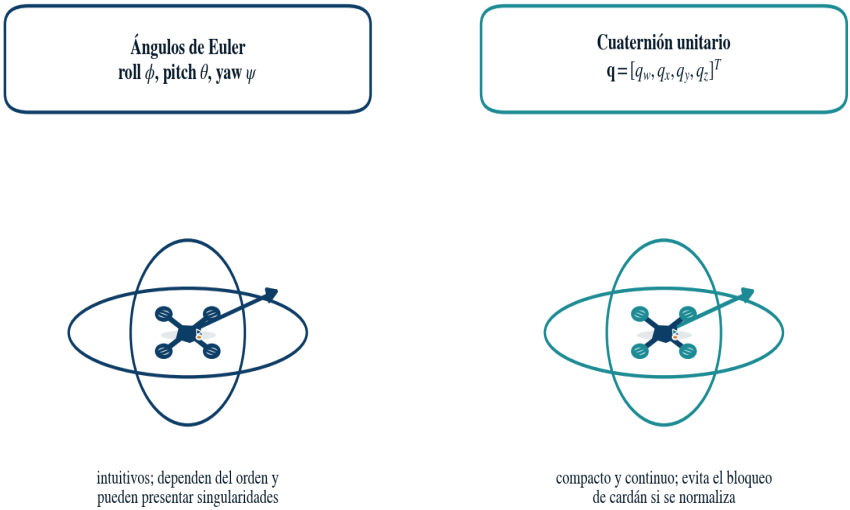


Tabla 4. *Representaciones de orientación*

Representación	Parámetros	Ventaja	Precaución
Ángulos de Euler	3	Interpretación directa	Singularidad y dependencia del orden
Matriz de rotación	9 con 6 restricciones	Operaciones geométricas simples	Redundancia y necesidad de preservar ortogonalidad
Cuaternión unitario	4 con 1 restricción	Sin singularidad, interpolación suave	Doble representación: $\mathbf{q}$ y $-\mathbf{q}$ describen la misma orientación

Idea clave. La representación se selecciona por la operación que se necesita. Los ángulos de Euler son útiles para explicar; las matrices para transformar; y los cuaterniones, para optimizar orientaciones sin singularidades.

2.5 Transformaciones homogéneas y puntos de interés

Una transformación homogénea combina rotación y traslación:

$$ {}^{I}\mathbf{T}_{B} = \begin{bmatrix} {}^{I}\mathbf{R}_{B} & {}^{I}\mathbf{p}_{B} \\ \mathbf{0}^{T} & 1 \end{bmatrix} $$

En un multirrotor es frecuente controlar un punto diferente al centro de masa: una cámara, una herramienta o un sensor. Si ese punto está desplazado una distancia (a, b, c) en el marco del cuerpo, su posición incorpora la rotación del vehículo. Esta idea aparece en los modelos cinemáticos de los seis trabajos y será desarrollada en el capítulo 3.

2.6 Ecuaciones diferenciales y tiempo discreto

La dinámica se formula habitualmente en tiempo continuo:

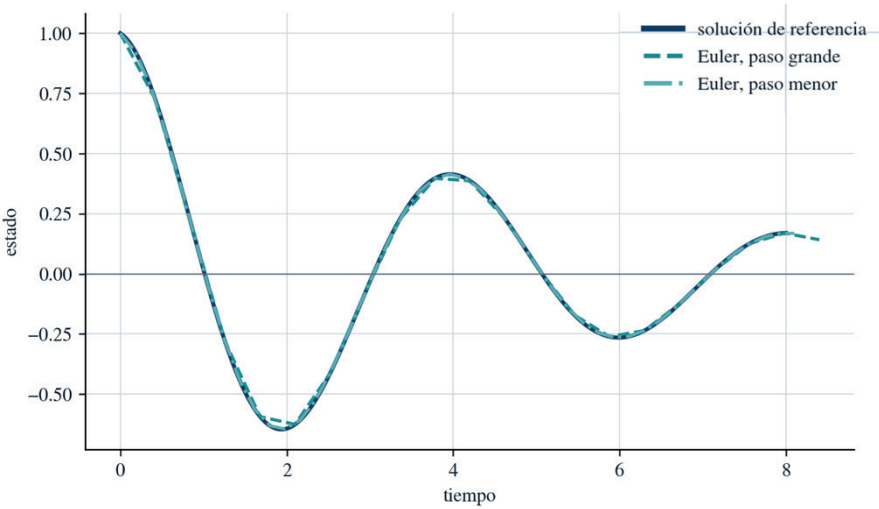
$$ \dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{x}(t), \mathbf{u}(t)\big) $$

Un computador, en cambio, actualiza valores en instantes separados por un período de muestreo Δt. La integración de Euler aproxima:

$$ \mathbf{x}_{k+1} = \mathbf{x}_{k} + \Delta t\, \mathbf{f}(\mathbf{x}_{k}, \mathbf{u}_{k}) $$

Figura 6.

Efecto conceptual del método de integración sobre la precisión y la estabilidad numérica



Es simple, pero puede introducir error o inestabilidad si Δt es grande. Runge–Kutta de cuarto orden evalúa la función varias veces y consigue mayor precisión. En los trabajos con NMPC, la elección del integrador modifica tanto la calidad de la predicción como el tiempo de cálculo. La Figura 6 ilustra el compromiso entre precisión y estabilidad numérica.

Advertencia. Reducir el período de muestreo no siempre mejora el sistema completo. Aumenta la frecuencia requerida y puede dejar menos tiempo para resolver el problema de optimización. El período debe ser compatible con la dinámica, los sensores y el hardware.

2.7 Ejemplo de transformación de velocidad

Considérese un vehículo con guiñada $\psi = 90^\circ$ que recibe una velocidad longitudinal de $1\,\mathrm{m/s}$ en su propio marco. En el marco inercial, esa orden no produce un movimiento en x, sino en y. Al aplicar $\mathbf{R}_z(90^\circ)$, el vector $[1, 0, 0]^T$ se transforma aproximadamente en $[0, 1, 0]^T$. Este ejemplo sencillo muestra por qué el jacobiano del capítulo siguiente depende de la orientación.

2.8 Resumen del capítulo

Los marcos de referencia determinan el significado de posiciones y velocidades. Las matrices de rotación, los ángulos de Euler y los cuaterniones son herramientas complementarias. La implementación digital exige discretizar ecuaciones diferenciales y equilibrar precisión con tiempo de cálculo.

2.9 Preguntas de repaso

1. ¿Qué diferencia existe entre una velocidad expresada en el marco inercial y una en el del cuerpo?
2. ¿Por qué la inversa de una matriz de rotación es su transpuesta?
3. ¿Qué problema evitan los cuaterniones?
4. ¿Qué representa el desplazamiento (a, b, c) de un punto de interés?
5. ¿Por qué un integrador más preciso puede aumentar el costo computacional del NMPC?

3. Modelación cinemática del multirrotor

Objetivos del capítulo. Derivar el modelo cinemático reducido, interpretar el jacobiano, incorporar un punto de interés, diferenciar seguimiento de camino y de trayectoria y preparar el modelo para su integración con la dinámica y el control. El capítulo separa la geometría de la respuesta dinámica, distinción que evita atribuir al controlador errores que pertenecen al modelo.

3.1 Pose reducida y velocidades de maniobrabilidad

En una plataforma comercial estabilizada internamente, una representación útil del estado de alto nivel es:

$$\boldsymbol{\eta} = [x \quad y \quad z \quad \psi]^T$$

El vector de velocidades de maniobrabilidad se expresa en el marco del cuerpo:

$$\boldsymbol{\nu} = [\nu_l \quad \nu_m \quad \nu_n \quad \omega]^T,$$

donde ν_l, ν_m y ν_n son velocidades longitudinal, lateral y vertical, y ω es la velocidad de guiñada. Esta elección coincide con las interfaces de control utilizadas en los casos DJI Matrice.

3.2 Posición de un punto de interés

Sea (x_0, y_0, z_0) la posición del centro de masa, para un punto desplazado (a, b, c) en el cuerpo, la pose puede escribirse como:

$$\begin{aligned} x &= x_0 + a\cos\psi - b\sin\psi, \\ y &= y_0 + a\sin\psi + b\cos\psi, \\ z &= z_0 + c, \\ \eta_\psi &= \psi \end{aligned}$$

La derivada muestra que una rotación de guiñada produce movimiento lineal cuando el punto no coincide con el centro de masa. Este término es importante para cámaras y herramientas: incluso si el centro de masa permanece casi fijo, la cámara puede describir un arco.

3.3 Jacobiano cinemático

Al derivar la pose se obtiene:

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(\psi)\boldsymbol{\nu},$$

con

$$\mathbf{J}(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 & -(a\sin\psi + b\cos\psi) \\ \sin\psi & \cos\psi & 0 & (a\cos\psi - b\sin\psi) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

La matriz cumple dos funciones. Primero, rota las velocidades lineales del cuerpo al marco inercial. Segundo, incorpora el desplazamiento del punto de interés. La Figura 7 resume este flujo.

Figura 7.
Flujo de la modelación cinemática: de velocidades del cuerpo a evolución de la pose



3.4 Aceleración cinemática

Si el controlador utiliza un modelo dinámico de velocidad, también se necesita la segunda derivada:

$$\ddot{\boldsymbol{\eta}} = \mathbf{J}(\psi)\dot{\boldsymbol{\nu}} + \dot{\mathbf{J}}(\psi, \omega)\boldsymbol{\nu}.$$

El primer término transforma aceleraciones del cuerpo. El segundo recoge el efecto de que el propio jacobiano cambia cuando el vehículo gira. Ignorar $\dot{\mathbf{J}}$ puede ser aceptable en movimientos lentos, pero introduce error en maniobras rápidas o cuando el punto de interés está lejos del centro de masa.

3.5 Inversión cinemática

Si se conoce una velocidad deseada de la pose, una referencia de maniobrabilidad puede obtenerse mediante la pseudoinversa:

$$\mathbf{v}_c = \mathbf{J}^{\dagger}(\psi)(\dot{\boldsymbol{\eta}}_d + \mathbf{v}_e),$$

donde $\mathbf{v}_e$ corrige el error de posición. Cuando $\mathbf{J}$ es cuadrada y no singular, $\mathbf{J}^{\dagger} = \mathbf{J}^{-1}$. La pseudoinversa mantiene la formulación general para sistemas con distinta cantidad de variables de tarea y control.

En los controladores estudiados se utiliza una función hiperbólica para limitar suavemente la corrección:

$$\mathbf{v}_e = \mathbf{K}_2 \tanh(\mathbf{K}_2^{-1}\mathbf{K}_1\tilde{\boldsymbol{\eta}}),$$

con $\tilde{\boldsymbol{\eta}} = \boldsymbol{\eta}_d - \boldsymbol{\eta}$. Cerca del origen, la función se comporta casi linealmente; lejos del origen, se satura y evita referencias excesivas.

Idea clave. La saturación suave no sustituye a todas las restricciones, pero protege a un controlador de realimentación frente a errores iniciales grandes y produce órdenes continuas.

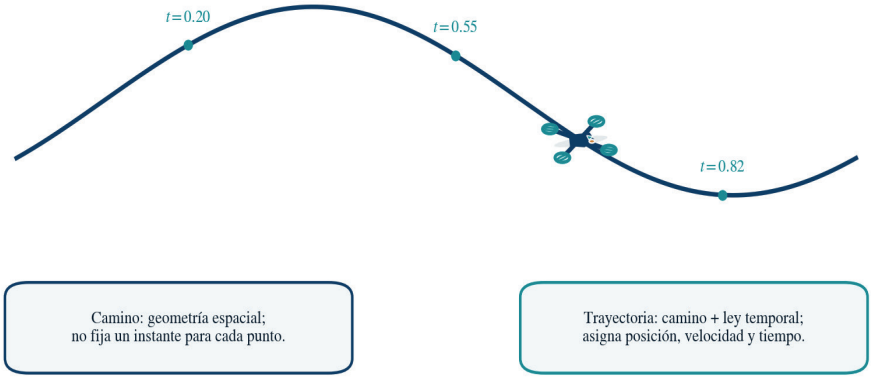
3.6 Camino y trayectoria

Una trayectoria asigna un instante a cada punto: $\mathbf{p}_d(t)$. Un camino define solo una geometría: $\mathbf{p}_d(\theta)$, donde θ indica avance. Esta diferencia será esencial en el control predictivo de contorno. Si aparece un obstáculo, un seguidor de trayectoria intenta recuperar tanto la

posición como el tiempo perdido; en cambio, un seguidor de camino puede reducir temporalmente el avance y conservar la geometría global. La Figura 8 contrasta ambos conceptos.

Figura 8.

Diferencia entre camino y trayectoria



3.7 Ejemplo numérico

Considérese $a = 0.2\,\text{m}$, $b = 0$, $\psi = 45°$, $v_l = 1\,\text{m/s}$ y $\omega = 0.5\,\text{rad/s}$. La velocidad en x incluye $v_l \cos\psi$ y un término por rotación, $-a\sin\psi\,\omega$. La contribución principal es aproximadamente $0.707\,\text{m/s}$ y la rotacional resta cerca de $0.071\,\text{m/s}$. La cámara se desplaza, por tanto, a unos $0.636\,\text{m/s}$ en x. Este valor sería diferente si se controlara el centro de masa.

3.8 Guía de implementación general con apoyo de inteligencia artificial (IA)

Esta guía puede implementarse en el entorno de preferencia siempre que se mantengan explícitos los marcos de referencia, las unidades y el orden de las variables. Una herramienta de IA puede generar una primera versión del código, pero no debe decidir por sí sola la convención geométrica.

Paso 1. Definir el vector de pose, el vector de velocidades y la convención de rotación. Es recomendable escribir una ficha breve que indique qué magnitud está expresada en el marco inercial y cuál en el marco del cuerpo.

Paso 2. Implementar la matriz de rotación y el jacobiano como funciones puras: una misma entrada debe producir siempre la misma salida y no debe depender de variables globales.

Paso 3. Crear pruebas simples antes de integrar el controlador. Con guiñada cero, una velocidad longitudinal debe producir movimiento sobre el eje inercial correspondiente; con 90 grados, debe aparecer sobre el eje ortogonal.

Paso 4. Integrar el modelo con un método numérico declarado y comparar, para un mismo período, los resultados de Euler y Runge-Kutta. La IA puede ayudar a escribir las pruebas, generar casos límite y explicar discrepancias, pero las tolerancias se fijan con criterio físico.

Paso 5. Registrar pose, velocidad, referencia y tiempo en un formato tabular. Esta interfaz será reutilizada por la identificación y los controladores de los capítulos posteriores.

3.9 Resumen del capítulo

El modelo cinemático transforma velocidades de maniobrabilidad en evolución de la pose. El jacobiano depende de la guiñada y del punto de interés. Su pseudoinversa permite construir controladores sencillos, mientras que su derivada conecta la cinemática con modelos dinámicos y predictivos.

3.10 Preguntas de repaso

1. ¿Qué variables contiene la pose reducida?
2. ¿Por qué el jacobiano depende de ψ?
3. ¿Qué efecto produce un punto de interés desplazado cuando el vehículo gira?
4. ¿Para qué se utiliza la pseudoinversa?
5. Explique con sus palabras la diferencia entre seguimiento de camino y seguimiento de trayectoria.

Capítulo II



Modelación e identificación


Z
X

4. Modelación dinámica y modelos reducidos

Objetivos del capítulo. Relacionar la velocidad de los rotores con fuerzas y momentos, comprender la estructura de los modelos Newton-Euler y Euler-Lagrange, derivar una representación reducida compatible con plataformas comerciales y reconocer la aparición de dinámica no modelada. La representación se elegirá por las señales realmente accesibles, de modo que el modelo conserve interpretación sin prometer información que el SDK no entrega.

4.1 De la velocidad del rotor al movimiento

Cada hélice genera una fuerza de empuje aproximadamente proporcional al cuadrado de su velocidad angular. En una forma simplificada,

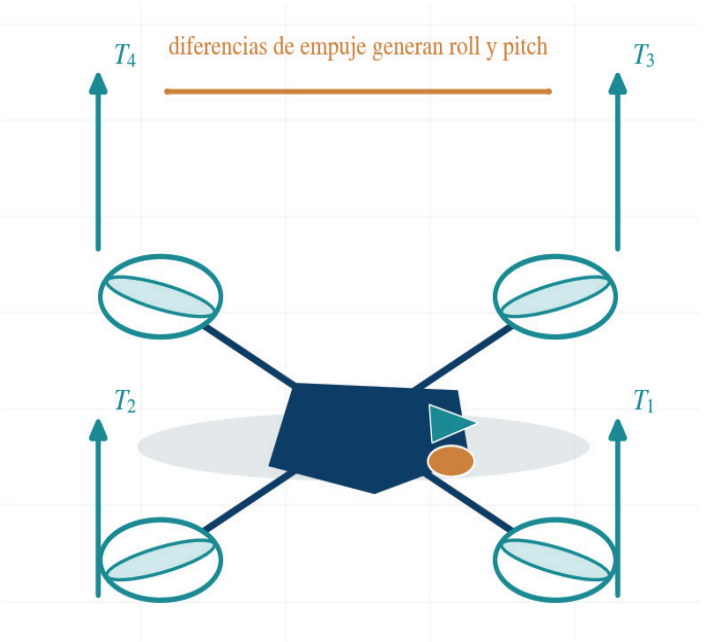
$$T_i = k_f \omega_i^2, \qquad \tau_i = k_m \omega_i^2,$$

donde k_f y k_m dependen de la hélice, el motor y la densidad del aire. La suma de los empujes produce la fuerza vertical del vehículo. Las diferencias entre rotores generan momentos de alabeo y cabeceo; la combinación de pares de reacción produce guiñada. La Figura 9

muestra cómo las diferencias de velocidad entre rotores generan fuerzas y momentos.

Figura 9.

Fuerzas y momentos producidos por diferencias de velocidad entre rotores



Un modelo de bajo nivel puede utilizar directamente ω_i como entrada. No obstante, en una plataforma comercial, el SDK suele ocultar esa capa. El usuario ordena velocidades lineales y angulares, mientras el controlador interno determina los comandos de motor. Por ello, este libro distingue entre entrada física (voltaje, velocidad de rotor, empuje o torque) y entrada accesible (velocidad de maniobrabilidad o referencia de pose).

4.2 Dinámica traslacional y rotacional

En una descripción Newton-Euler, la traslación del centro de masa satisface

$$m\ddot{\mathbf{p}} = \mathbf{R}(\mathbf{q})\begin{bmatrix}0\\0\\T\end{bmatrix} - mg\mathbf{e}_3 + \mathbf{f}_d + \mathbf{f}_{ext},$$

donde m es la masa; T, el empuje total; $\mathbf{f}_d$, el arrastre; y $\mathbf{f}_{ext}$, una perturbación. La rotación puede expresarse como:

$$\mathbf{J}_B\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbf{J}_B\boldsymbol{\omega}) = \boldsymbol{\tau} + \boldsymbol{\tau}_g + \boldsymbol{\tau}_{ext}.$$

Estas ecuaciones muestran los principales acoplamientos. La dirección del empuje depende de la orientación; la aceleración angular depende de la inercia; y los términos cruzados aumentan con la velocidad. También revelan por qué un cambio de carga útil altera el comportamiento: modifica la masa, el centro de masa y, con frecuencia, la matriz de inercia.

4.3 Formulación de Euler-Lagrange

La formulación de Euler-Lagrange parte de la energía cinética K y la energía potencial P. Con el lagrangiano $L = K - P$, la ecuación general es:

$$\frac{d}{dt}\frac{\partial L}{\partial \mathbf{q}_{EL}} - \frac{\partial L}{\partial \mathbf{q}_{EL}} = \boldsymbol{\tau}$$

Su ventaja es sistematizar la derivación y producir una estructura reconocible. En esta sección, $\mathbf{q}_{EL}$ denota coordenadas generalizadas y se distingue del cuaternión unitario $\mathbf{q}$ empleado para representar actitud.

$$\mathbf{M}(\mathbf{q}_{EL})\ddot{\mathbf{q}}_{EL} + \mathbf{C}(\mathbf{q}_{EL}, \dot{\mathbf{q}}_{EL})\dot{\mathbf{q}}_{EL} + \mathbf{g}(\mathbf{q}_{EL}) = \boldsymbol{\tau}$$

La matriz **M** contiene masa e inercia; **C** reúne términos de Coriolis y centrípetos; y **g** incorpora gravedad. En el hexacóptero estudiado por Recalde *et al.* (2022a), la formulación se adapta para incluir el efecto de controladores proporcional-integral-derivativo (PID) internos y obtener una relación en el espacio de velocidades.

4.4 El papel de los lazos internos

Un SDK comercial puede recibir una referencia $\boldsymbol{\mu}_{ref}$ y entregar como respuesta una velocidad real $\boldsymbol{\nu}$. Entre ambas existe una dinámica cerrada que incluye motores, controladores PID, saturaciones, filtros y aerodinámica. En lugar de ignorar esa dinámica, se incorpora en un modelo reducido:

$$\boldsymbol{\mu}_{ref}(t) = \mathbf{M}(\boldsymbol{\xi}, a, b)\dot{\boldsymbol{\nu}}(t) + \mathbf{C}(\boldsymbol{\xi}, \boldsymbol{\nu}, a, b)\boldsymbol{\nu}(t)$$

El vector $\boldsymbol{\xi}$ agrupa parámetros físicos, mecánicos, eléctricos y de control interno. En los casos de estudio se emplea una parametrización de 19 valores. No todos poseen una interpretación física aislada: algunos representan combinaciones equivalentes que solo pueden identificarse como conjunto. La Figura 10 relaciona los tres niveles de representación y la Tabla 5 resume su uso recomendado.

Figura 10.
Tres niveles complementarios de representación: físico, reducido y basado en datos



Tabla 5. *Niveles de modelación y uso recomendado*

Nivel	Entrada típica	Fortalezas	Uso principal
Físico completo	Empujes y torques	Interpretabilidad, análisis de energía y estabilidad	Diseño de plataforma y control de bajo nivel
Reducido	Velocidades de referencia	Compatible con SDK, menor dimensión	Control de trayectoria en plataformas comerciales
Basado en datos	Entradas y estados medidos	Ajuste directo a la respuesta real	Identificación, adaptación y corrección de discrepancias

4.5 Arrastre, incertidumbre y dinámica no modelada

El arrastre aerodinámico suele aproximarse como una fuerza opuesta a la velocidad. Una forma lineal es $\mathbf{f}_d = -\mathbf{K}_d\mathbf{v}$; para velocidades mayores puede ser necesario un término cuadrático. Sin embargo, incluso un modelo detallado omite fenómenos: ráfagas, efecto suelo, flexibilidad, retardos, error de calibración, variación de batería o movimiento de una carga.

Para representar esa diferencia se añade un término desconocido:

$$\boldsymbol{\mu}_T = \mathbf{M}_n\dot{\boldsymbol{\nu}} + \mathbf{C}_n\boldsymbol{\nu} + \boldsymbol{\mu}_u$$

El subíndice n indica el modelo nominal y $\boldsymbol{\mu}_u$, la contribución no modelada. La pregunta deja de ser ¿cómo obtener un modelo perfecto? y pasa a ser ¿cómo mantener un desempeño aceptable cuando el modelo es imperfecto? Los capítulos 6 y 9 responden mediante actualización en línea, leyes adaptativas y observadores.

Idea clave. Todo modelo es una aproximación. La robustez no consiste en negar el error de modelación, sino en medirlo, acotarlo o estimarlo para que el controlador pueda reaccionar.

4.6 Elección de estados y entradas

La selección de variables determina la complejidad. Un modelo completo puede incluir posición, velocidad, cuaternión, velocidad angular y dinámica de actuadores. Un modelo reducido puede limitarse a $\boldsymbol{\eta}$ y $\boldsymbol{\nu}$. Para un NMPC de alto nivel, esta reducción disminuye el número de variables de decisión y facilita la ejecución en tiempo real.

La reducción debe respetar la misión. Si se requiere vuelo agresivo con orientación libre, no es razonable conservar solo la guiñada. Si la cámara está rígidamente unida y el vehículo opera con inclinaciones pequeñas, un modelo reducido puede ser suficiente. El trabajo de control de contorno de Guevara *et al.* (2025) vuelve a incorporar el cuaternión porque necesita alinear la actitud con la dirección del camino y evitar singularidades.

4.7 Ejemplo de discrepancia de modelo

Supóngase que el modelo fue identificado con una masa de 2.5 kg y luego se instala un sensor de 0.4 kg. La misma referencia vertical produce una aceleración menor. Un controlador puramente nominal acumulará error hasta que su realimentación compense. Un esquema adaptativo puede estimar una corrección equivalente; una descomposición en modos dinámicos con control (DMDc) en línea puede actualizar la matriz de entrada; y un observador puede inferir una perturbación persistente.

4.8 Guía de implementación general con apoyo de IA

La implementación dinámica debe comenzar por el modelo más sencillo que represente la entrada realmente disponible. En una plataforma comercial con SDK de velocidad, no es necesario reconstruir de inmediato los torques de cada motor.

Paso 1. Enumerar parámetros físicos, variables medidas y entradas disponibles. Marcar cuáles son conocidos, cuáles se estimarán y cuáles se agruparán en términos equivalentes.

Paso 2. Implementar por separado los términos de inercia, Coriolis, gravedad y arrastre. Esta separación facilita detectar signos incorrectos y permite sustituir un término sin reescribir el modelo completo.

Paso 3. Verificar equilibrio, simetría y unidades. En vuelo estacionario, las aceleraciones previstas deben ser compatibles con cero; una masa duplicada no puede producir la misma aceleración ante la misma fuerza.

Paso 4. Comparar el modelo físico con un modelo reducido usando exactamente las mismas entradas y condiciones iniciales. La IA puede traducir ecuaciones simbólicas a funciones numéricas y proponer pruebas, pero los resultados deben contrastarse con leyes físicas y datos.

Paso 5. Conservar una interfaz común $f(x, u, \text{parámetros})$ para que el mismo integrador pueda utilizarse en simulación, identificación y control predictivo.

4.9 Resumen del capítulo

La dinámica física relaciona empujes y momentos con aceleraciones. En plataformas comerciales resulta útil un modelo reducido que

incluya la acción de los lazos internos. Esta representación facilita el control de alto nivel, pero concentra fenómenos desconocidos en parámetros y términos no modelados.

4.10 Preguntas de repaso

1. ¿Por qué el empuje suele aproximarse como proporcional a ω_i^2?
2. ¿Qué información contiene la matriz de inercia?
3. ¿Qué representa la matriz $\mathbf{C}$ en una formulación Euler-Lagrange?
4. ¿Por qué un modelo reducido puede ser más útil que uno completo para un SDK comercial?
5. Mencione cuatro fuentes de dinámica no modelada.

5. Identificación de sistemas multirrotor

Objetivos del capítulo. Formular el problema de identificación, diseñar datos informativos, diferenciar identificación paramétrica y DMDc, aplicar validación independiente y utilizar métricas para seleccionar un modelo destinado al control. El criterio de selección no será únicamente el ajuste sobre los datos de entrenamiento, sino la capacidad de predecir un ensayo independiente dentro del régimen de operación previsto.

5.1 Identificar no es ajustar una curva cualquiera

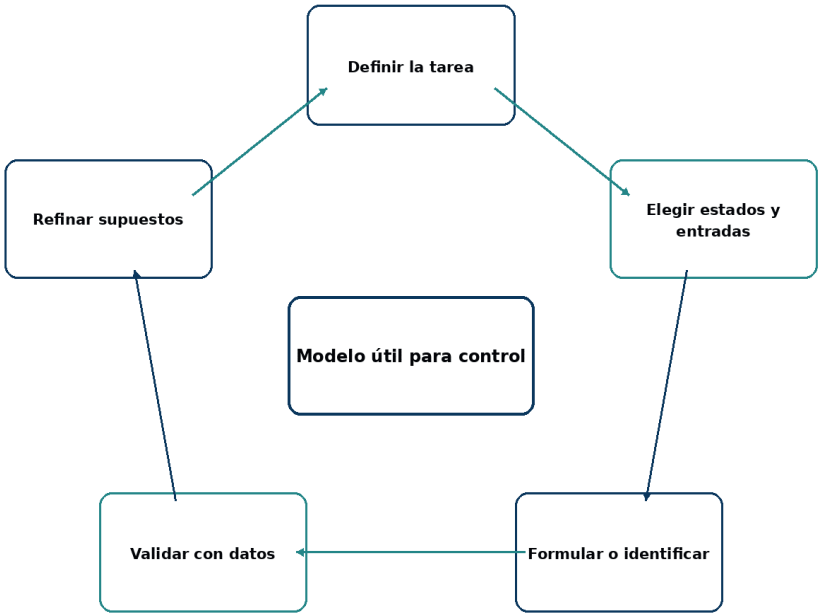
La identificación busca un modelo capaz de predecir la respuesta del sistema ante entradas relevantes. No basta con minimizar el error sobre los mismos datos utilizados para estimar parámetros. Un modelo

útil debe generalizar a señales, velocidades y condiciones no vistas durante el ajuste.

El proceso comienza definiendo qué se desea predecir. Para los modelos reducidos de este libro, las entradas son referencias de velocidad, y las salidas son velocidades reales o sus derivadas. Después se diseña una excitación que active cada canal sin comprometer la seguridad. Finalmente, los datos se separan en conjuntos de identificación y validación. La Figura 11 resume este ciclo iterativo de modelación.

Figura 11.

Modelación: proceso iterativo que alterna hipótesis, datos, validación y refinamiento



5.2 Diseño de la excitación

Una entrada constante revela el régimen estacionario, pero contiene poca información dinámica. Para identificar tiempos de respuesta y acoplamientos se utilizan escalones, señales pseudoaleatorias, *chirps* o trayectorias que cambian velocidad y dirección. La señal debe cubrir el rango de operación previsto. Un modelo identificado solo cerca del vuelo estacionario no puede evaluarse con justicia en una maniobra agresiva.

El período de muestreo debe ser lo bastante pequeño para observar la dinámica. También es necesario sincronizar entrada y salida: un desfase de pocos pasos puede parecer una constante de tiempo inexistente. El filtrado debe aplicarse con cuidado, porque reducir ruido también puede alterar fase y amplitud.

Advertencia. La derivación numérica amplifica ruido. Cuando DMDc utiliza la derivada de la velocidad, conviene filtrar, emplear diferencias robustas o formular una versión discreta con estados consecutivos.

5.3 Identificación paramétrica

Si el modelo puede escribirse de forma lineal en sus parámetros,

$$\boldsymbol{\mu}_{ref}(t) = \boldsymbol{\Theta}(\dot{\mathbf{v}}, \mathbf{v})\xi,$$

la matriz regresora $\boldsymbol{\Theta}$ se construye con datos medidos y el vector $\boldsymbol{\xi}$ se estima minimizando

$$V_m(\boldsymbol{\xi}) = \sum_{k=1}^{N_d} \left\| \boldsymbol{\mu}_{ref,k} - \boldsymbol{\Theta}_k \boldsymbol{\xi} \right\|_{\mathbf{Q}_m}^2$$

Puede resolverse mediante mínimos cuadrados o programación cuadrática secuencial (Sequential Quadratic Programming, SQP)

cuando existen límites o relaciones adicionales. La ventaja principal es conservar la estructura del modelo. Su limitación es que la regresora debe estar bien condicionada y que una estructura física incompleta no desaparece por estimar mejor sus parámetros.

5.4 Dynamic Mode Decomposition with control (DMDc)

DMDc aproxima la evolución mediante la fórmula:

$$\dot{\boldsymbol{\nu}} = \mathbf{A}\boldsymbol{\nu} + \mathbf{B}\boldsymbol{\mu}_{ref}$$

Con s mediciones se forman las matrices:

$$\dot{\mathbf{X}} = [\dot{\boldsymbol{\nu}}_1 \quad \cdots \quad \dot{\boldsymbol{\nu}}_s], \quad \mathbf{X} = [\boldsymbol{\nu}_1 \quad \cdots \quad \boldsymbol{\nu}_s],$$

$$\mathbf{U} = [\boldsymbol{\mu}_{ref,1} \quad \cdots \quad \boldsymbol{\mu}_{ref,s}]$$

Al definir $\boldsymbol{\Omega} = [\mathbf{X}^T \ \mathbf{U}^T]^T$, la solución de mínimos cuadrados es

$$[\mathbf{A} \quad \mathbf{B}] \approx \dot{\mathbf{X}}\boldsymbol{\Omega}^{\dagger}$$

La pseudoinversa se calcula de manera estable mediante descomposición en valores singulares. El truncamiento de valores pequeños reduce ruido y evita amplificar direcciones poco excitadas. La Figura 12 muestra el flujo completo de identificación DMDc y la Tabla 6 compara los métodos empleados.

Figura 12.

Proceso de identificación DMDc desde datos de vuelo hasta un modelo lineal controlado

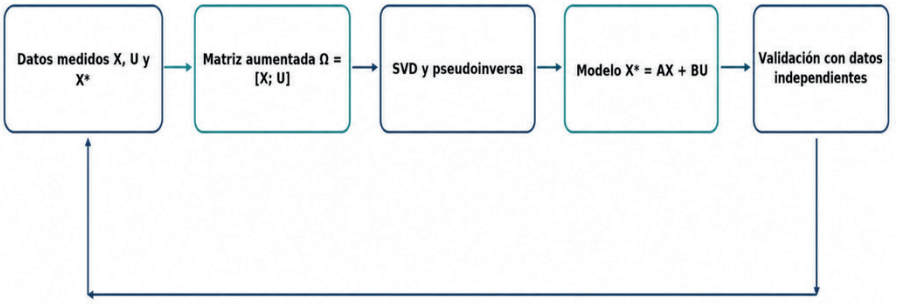


Tabla 6. *Comparación de métodos de identificación empleados en los casos de estudio*

Método	**Estructura previa**	**Resultado**	**Ventaja**	**Riesgo**
Mínimos cuadrados/SQP	Modelo lineal en parámetros	$\boldsymbol{\xi}$	Interpretabilidad y restricciones	Condicionamiento y sesgo estructural
DMDc fuera de línea	Estados y entradas medidos	$\mathbf{A}, \mathbf{B}$	Implementación directa y bajo orden	Dependencia del conjunto de datos
DMDc en línea	Ventana reciente de estados y entradas	$\mathbf{A}_k, \mathbf{B}_k$	Adaptación a cambios	Sensibilidad al ruido y falta de excitación
Ley adaptativa	Regresor de incertidumbre	$\hat{\boldsymbol{\xi}}_u$	Compensación con análisis de estabilidad	Convergencia condicionada por excitación
Observador de momento	Modelo nominal y mediciones	Perturbación estimada	Actúa como sensor virtual	Ajuste de ganancias y filtrado

5.5 Validación independiente

La validación compara la salida medida con la predicción del modelo usando entradas distintas de las empleadas en identificación. Las métricas frecuentes son la raíz del error cuadrático medio (RMSE), la norma del error, la integral del error cuadrático (ISE) y el porcentaje de ajuste. También deben revisarse residuos: si conservan patrones, existe dinámica sin representar.

En el hexacóptero DJI Matrice 600, Recalde *et al.* (2022a) comparó la formulación Euler-Lagrange identificada mediante programación cuadrática secuencial (SQP) con descomposición en modos dinámicos (DMD). El DMD obtuvo menor ISE en los canales longitudinal, lateral y angular y un valor prácticamente igual en el canal vertical. Esta evidencia justificó seleccionar el modelo DMD para el NMPC. La Figura 13 representa los errores reportados y la Tabla 7 presenta sus valores por canal.

Figura 13.

Error integral cuadrático reportado para los modelos Euler-Lagrange y DMD del hexacóptero

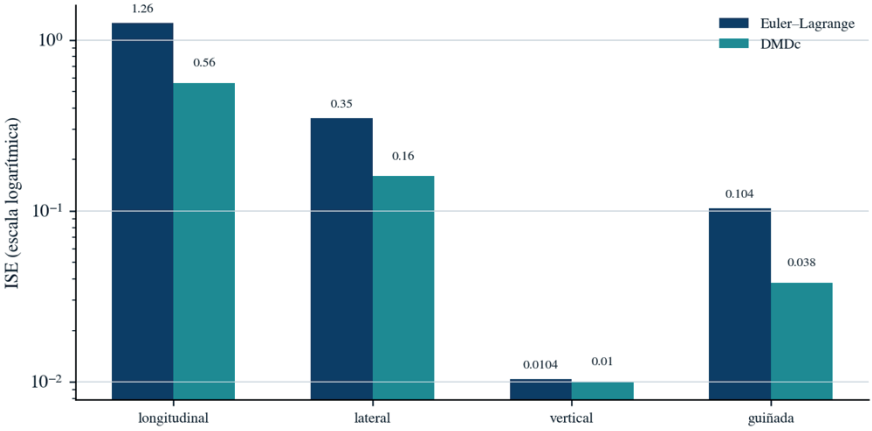


Tabla 7. *Error ISE de los modelos identificados en el hexacóptero*

Canal de velocidad	Euler-Lagrange + SQP	DMD	Reducción aproximada
Longitudinal	1.26	0.56	55.6 %
Lateral	0.35	0.16	54.3 %
Vertical	0.0104	0.0100	3.8 %
Guiñada	0.104	0.038	63.5 %

Los valores no prueban que DMD sea siempre superior. Muestran que, para esos datos, esa estructura describió mejor la dinámica de alto nivel. En otro régimen, con un conjunto de excitación distinto o con interés en parámetros físicos, la conclusión podría cambiar.

5.6 Selección de rango mediante descomposición en valores singulares (SVD)

La SVD descompone $\mathbf{\Omega} = \mathbf{U\Sigma V}^T$. Los valores singulares grandes representan direcciones dominantes; los muy pequeños pueden corresponder a ruido o a variables casi dependientes. El rango truncado r equilibra fidelidad y robustez.

Una estrategia práctica consiste en observar la energía acumulada, validar varios rangos y elegir el menor que preserve la predicción. Un rango demasiado bajo elimina dinámica útil; uno demasiado alto reproduce ruido y empeora la estabilidad numérica.

5.7 Identificabilidad y excitación persistente

No todos los parámetros pueden estimarse con cualquier trayectoria. Si dos columnas de la regresora son casi proporcionales, sus efectos no pueden separarse. La excitación persistente exige que las entradas

recorran suficientes direcciones del espacio. En un vuelo seguro, esto se logra mediante secuencias diseñadas y no necesariamente a través de maniobras extremas.

Ejemplo. Para identificar el acoplamiento entre velocidad lateral y guiñada, una trayectoria que solo sube y baja es insuficiente. Deben incluirse desplazamientos laterales y rotaciones con amplitudes y frecuencias compatibles con el régimen de uso.

5.8 Guía de implementación general con apoyo de IA

La identificación se implementa como una cadena reproducible: adquisición, limpieza, construcción de matrices, estimación y validación independiente. El algoritmo no debe recibir datos de validación durante el ajuste.

Paso 1. Diseñar señales de excitación seguras que activen todos los canales relevantes sin saturar la plataforma. Registrar la referencia enviada y la respuesta realmente medida.

Paso 2. Sincronizar, filtrar y normalizar los datos sin alterar su significado físico. Toda transformación debe guardarse junto con sus parámetros para aplicarla después a datos nuevos.

Paso 3. Construir las matrices de instantáneas X, U y X^+ y verificar sus dimensiones. La IA puede generar rutinas de comprobación y mensajes de error descriptivos.

Paso 4. Estimar el modelo, documentar el rango retenido y guardar valores singulares, matrices identificadas y versión del conjunto de datos.

Paso 5. Evaluar el modelo con una secuencia no utilizada en el ajuste. Reportar error por canal, sesgo, máximos y comportamiento temporal; no limitar la validación a una única gráfica.

5.9 Resumen del capítulo

La identificación convierte datos de entrada y salida en un modelo predictivo. La estimación paramétrica conserva estructura; DMDc obtiene matrices directamente de datos. La selección debe basarse en validación independiente, métricas y análisis de residuos, no solo en el error de ajuste.

5.10 Preguntas de repaso

1. ¿Por qué se separan datos de identificación y validación?
2. ¿Qué función cumple una señal de excitación?
3. ¿Qué matrices estima DMDc?
4. ¿Qué problema resuelve el truncamiento de SVD?
5. ¿Por qué un menor error de identificación no garantiza por sí solo un mejor controlador?

6. Identificación en línea y control adaptativo

Objetivos del capítulo. Comprender la necesidad de actualizar el modelo durante el vuelo, describir DMDc con ventana móvil, integrar el modelo en un controlador de dinámica inversa y reconocer las condiciones que permiten o impiden la adaptación. La actualización solo es útil cuando sus datos son informativos y cuando una política de aceptación impide sustituir un modelo válido por uno numéricamente frágil.

6.1 Cuándo un modelo fuera de línea deja de ser suficiente

Un modelo identificado antes del vuelo supone que la dinámica permanecerá aproximadamente constante. Esa hipótesis se debilita cuando cambia la carga útil, disminuye el voltaje de batería, aparece viento, se modifica el centro de masa o el vehículo entra en un régimen no observado. La discrepancia se manifiesta como error de seguimiento y acciones de control que llegan tarde o con amplitud incorrecta.

Actualizar el modelo en línea permite que el controlador utilice la información reciente. No se trata de reconstruir toda la física a cada instante, sino de ajustar una representación compacta que describa la relación local entre estados y entradas.

6.2 DMDc con ventana móvil

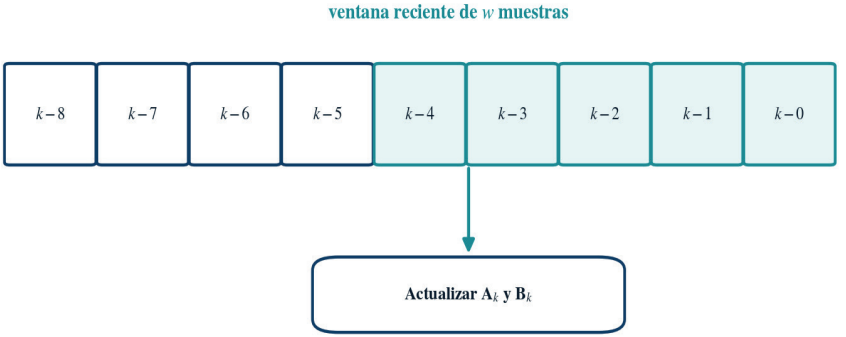
En tiempo discreto se considera

$$\mathbf{v}_{k+1} = \mathbf{A}_k \mathbf{v}_k + \mathbf{B}_k \boldsymbol{\mu}_k$$

Una ventana de longitud w conserva las últimas mediciones. Cuando llega un nuevo dato, se añade al extremo y se elimina el más antiguo. Las matrices $\mathbf{A}_k$ y $\mathbf{B}_k$ se recalculan o se actualizan mediante fórmulas recursivas. La ventana corta reacciona rápido, pero es sensible al ruido; la ventana larga produce una estimación más suave, aunque tarda en reflejar cambios. La Figura 14 ilustra el funcionamiento de la ventana móvil.

Figura 14.

Ventana móvil para actualizar un modelo DMDc con las mediciones más recientes



En el estudio de Guevara *et al.* (2023b) se utilizó una ventana de seis muestras dentro de un esquema Hardware-in-the-Loop ejecutado a 30 Hz. El modelo en línea mostró una reducción del error respecto al DMDc fijo y permitió compensar perturbaciones desconocidas en simulación.

6.3 Actualización recursiva

Recalcular una pseudoinversa completa en cada muestra puede ser costoso. Las actualizaciones recursivas aprovechan la estimación anterior. Una estructura genérica es:

$$\boldsymbol{\Theta}_k = \boldsymbol{\Theta}_{k-1} + \mathbf{K}_k(\boldsymbol{\nu}_{k+1} - \boldsymbol{\Theta}_{k-1}\boldsymbol{\varphi}_k),$$

donde $\boldsymbol{\Theta}_k = [\mathbf{A}_k\ \mathbf{B}_k]$, $\boldsymbol{\varphi}_k = \left[\boldsymbol{\nu}_k^T\ \boldsymbol{\mu}_k^T\right]^T$ y $\mathbf{K}_k$ es una ganancia dependiente de la información acumulada. Las fórmulas tipo Sherman-Morrison actualizan inversas o matrices de covarianza sin repetir toda la factorización.

6.4 Control cinemático y compensación dinámica

El controlador en cascada puede organizarse en dos niveles. El primero calcula una velocidad deseada mediante cinemática inversa:

$$\boldsymbol{\nu}_c = \mathbf{J}^{\dagger}[\dot{\boldsymbol{\eta}}_d + \mathbf{K}_2 \tanh(\mathbf{K}_2^{-1}\mathbf{K}_1\tilde{\boldsymbol{\eta}})]$$

El segundo usa el modelo identificado para generar la entrada:

$$\boldsymbol{\mu} = \mathbf{B}_k^{-1}(\boldsymbol{\sigma} - \mathbf{A}_k\boldsymbol{\nu}),$$

$$\boldsymbol{\sigma} = \dot{\boldsymbol{\nu}}_c + \mathbf{K}_4 \tanh(\mathbf{K}_4^{-1}\mathbf{K}_3\tilde{\boldsymbol{\nu}})$$

Si el modelo cambia, la compensación utiliza inmediatamente las matrices actualizadas. La estructura conserva un costo mucho menor que un NMPC, porque no resuelve un horizonte de optimización.

6.5 El problema de la falta de excitación

Durante un vuelo estacionario, las entradas y estados varían poco. La ventana reciente puede perder rango y el modelo se vuelve indeterminado o ruidoso. Por ello, se utilizan regularización, límites de actualización, factores de olvido y detección de condición numérica. Una opción es congelar el modelo cuando la información no es suficiente.

Advertencia. "Adaptativo" no significa siempre mejora. Una actualización basada en datos pobres puede deteriorar el modelo. El algoritmo debe vigilar rango, ruido, magnitud de parámetros y plausibilidad de la predicción.

6.6 Hardware-in-the-Loop (HIL) como etapa intermedia

En hardware en el lazo (Hardware-in-the-Loop, HIL), el código y el hardware de control interactúan con una planta simulada en tiempo real. Esta configuración permite evaluar tiempos de cómputo, comunicaciones y fallos sin arriesgar el vehículo. El trabajo de DMDc en

línea utiliza el SDK de DJI, Robot Operating System (ROS) y el simulador del fabricante para acceder a variables equivalentes a las del Matrice 100.

El valor de HIL no se limita a "ver si el dron sigue la trayectoria". Revela pérdida de paquetes, desfases, carga del procesador, problemas de unidades y comportamiento cuando la tasa del controlador no coincide con la del simulador.

6.7 Beneficios del modelo en línea

Hay tres beneficios principales. Primero, la ventana móvil reproduce mejor cambios de dinámica que un modelo fijo. Segundo, la compensación dinámica genera acciones más suaves y reduce el error frente a perturbaciones no conocidas por el modelo nominal. Tercero, la actualización puede ejecutarse dentro de una frecuencia de 30 Hz para un modelo de cuatro velocidades.

Estos beneficios dependen de la escala. Un modelo con muchas variables, restricciones o percepción pesada puede requerir otra estrategia. El propósito del capítulo es mostrar que la adaptación puede integrarse con una representación reducida y con un controlador analíticamente sencillo.

6.8 Guía de implementación general con apoyo de IA

El modelo en línea requiere separar claramente el ciclo rápido de control del proceso de actualización. La identificación no debe bloquear la generación de comandos.

Paso 1. Crear un búfer circular con marcas de tiempo para estados y entradas. El tamaño de ventana se expresa en muestras y en segundos.

Paso 2. Actualizar el modelo únicamente cuando exista suficiente excitación y las mediciones superen controles de calidad. Si no se cumple esta condición, se conserva el último modelo válido.

Paso 3. Limitar la variación permitida de las matrices identificadas y supervisar su condicionamiento. Una actualización numéricamente posible no siempre es físicamente razonable.

Paso 4. Ejecutar la actualización en un hilo o proceso separado y transferir el nuevo modelo mediante una operación atómica. La IA puede ayudar a adaptar este patrón al lenguaje seleccionado y a generar pruebas de concurrencia.

Paso 5. Comparar el error antes y después de cada actualización y almacenar una bandera que permita volver al modelo anterior cuando el desempeño empeore.

6.9 Resumen del capítulo

DMDc en línea actualiza matrices con una ventana de datos recientes. La selección de longitud de ventana controla el compromiso entre rapidez y ruido. Al combinar el modelo con cinemática inversa y compensación dinámica se obtiene un controlador adaptativo de bajo costo, cuya seguridad depende de vigilar la calidad de la identificación.

6.10 Preguntas de repaso

1. ¿Qué cambios de operación justifican actualizar un modelo en línea?
2. ¿Qué ocurre al utilizar una ventana demasiado corta?
3. ¿Qué ventaja tiene una actualización recursiva?

4. ¿Por qué el vuelo estacionario puede ser problemático para la identificación?
5. ¿Qué aspectos, además del seguimiento, deben verificarse en HIL?

Capítulo III



Control de vuelo y seguridad


Z
X

7. Controladores de realimentación

Objetivos del capítulo. Comprender la estructura de un lazo de realimentación, formular control cinemático y compensación dinámica, interpretar una demostración de estabilidad basada en Lyapunov y comparar estas alternativas con un controlador predictivo en términos de precisión y costo computacional. Las comparaciones se leerán con el mismo escenario, las mismas unidades y el mismo presupuesto temporal; de otro modo, una mejora aparente puede ser consecuencia de condiciones distintas.

7.1 El principio de realimentación

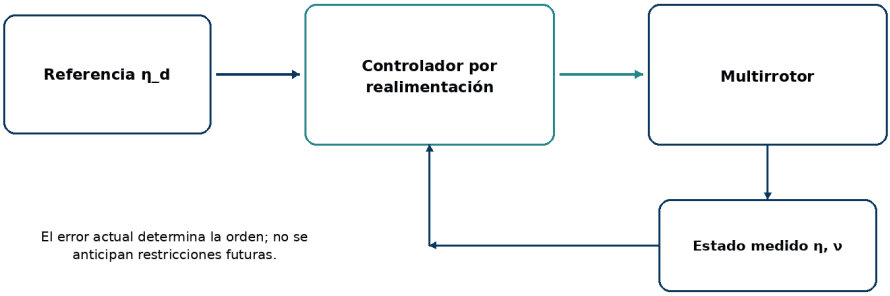
Un controlador por realimentación mide el estado actual, lo compara con una referencia y utiliza el error para calcular una acción. En su forma más sencilla es:

$$\tilde{\mathbf{x}} = \mathbf{x}_d - \mathbf{x}, \qquad \mathbf{u} = \mathcal{K}(\tilde{\mathbf{x}})$$

La realimentación es necesaria porque ningún modelo reproduce exactamente el vuelo. El viento, los retardos, las variaciones de batería y los errores de estimación hacen que una acción calculada solo a partir de una predicción abierta se desvíe de la realidad. Al medir nuevamente, el controlador corrige esa desviación. La Figura 15 representa este lazo básico de realimentación.

Figura 15.

Lazo básico de control por realimentación aplicado a un multirrotor



El diagrama contiene cuatro elementos. La referencia representa la misión; el controlador convierte el error en una orden; la planta incluye el vehículo y sus lazos internos; y los sensores reconstruyen las variables necesarias. Las perturbaciones pueden actuar sobre cualquiera de esos elementos. Por ello, una buena ley de control no se evalúa únicamente por su respuesta nominal, sino también por su comportamiento ante incertidumbre.

7.2 Control cinemático inverso

El controlador cinemático parte de la relación:

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(\psi)\boldsymbol{\nu}$$

Si se desea que la pose siga $\boldsymbol{\eta}_d(t)$, se calcula una velocidad de maniobrabilidad de referencia:

$$\boldsymbol{\nu}_c = \mathbf{J}^{\dagger}(\psi)[\dot{\boldsymbol{\eta}}_d + \mathbf{K}_2 \tanh(\mathbf{K}_2^{-1}\mathbf{K}_1\tilde{\boldsymbol{\eta}})]$$

La pseudoinversa realiza el mapeo entre el espacio de tarea y las velocidades del vehículo. El término hiperbólico corrige el error y, al mismo tiempo, limita suavemente la magnitud de la referencia. Cerca

del origen se aproxima a una ley proporcional; lejos del origen evita que la acción crezca sin límite.

Este controlador tiene tres ventajas: es intuitivo, requiere poco cálculo y puede implementarse con tasas muy altas. Su principal limitación es que supone que la velocidad ordenada se alcanza casi de inmediato. Cuando la dinámica interna es lenta, la trayectoria es rápida o existe una perturbación importante, aparece un error adicional.

7.3 Compensación dinámica inversa

Para incorporar la respuesta de velocidad se emplea el modelo reducido:

$$\boldsymbol{\mu} = \mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu}$$

Primero se obtiene $\boldsymbol{\nu}_c$ mediante la cinemática. Después se define el error de velocidad $\tilde{\boldsymbol{\nu}} = \boldsymbol{\nu}_c - \boldsymbol{\nu}$ y una aceleración virtual:

$$\boldsymbol{\sigma} = \dot{\boldsymbol{\nu}}_c + \mathbf{K}_4 \tanh(\mathbf{K}_4^{-1}\mathbf{K}_3\tilde{\boldsymbol{\nu}})$$

La orden se obtiene por inversión dinámica:

$$\boldsymbol{\mu}_c = \mathbf{M}\boldsymbol{\sigma} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu}$$

En un modelo DMDc equivalente, la operación puede escribirse como:

$$\boldsymbol{\mu}_c = \mathbf{B}^{\dagger}(\boldsymbol{\sigma} - \mathbf{A}\boldsymbol{\nu})$$

La compensación anticipa la aceleración necesaria para que la velocidad real siga a la deseada. Por eso suele superar al controlador puramente cinemático sin alcanzar el costo de un NMPC.

Ejemplo. Si el vehículo transporta una cámara adicional, la respuesta lateral puede volverse más lenta. El controlador cinemático mantiene la misma orden y espera que la velocidad se cumpla. La

compensación dinámica utiliza el modelo identificado para incrementar o reducir la referencia de forma coherente con esa nueva respuesta.

7.4 Lectura intuitiva de la estabilidad de Lyapunov

Una función de Lyapunov es una medida escalar de energía o distancia al objetivo. Para el error de pose puede elegirse:

$$V_k(\tilde{\boldsymbol{\eta}}) = \frac{1}{2}\tilde{\boldsymbol{\eta}}^T \boldsymbol{\Lambda}\tilde{\boldsymbol{\eta}}, \qquad \boldsymbol{\Lambda} > 0$$

Al sustituir la ley cinemática, su derivada toma la forma:

$$\dot{V}_k = -\tilde{\boldsymbol{\eta}}^T \boldsymbol{\Lambda}\mathbf{K}_2 \tanh(\mathbf{K}_2^{-1}\mathbf{K}_1\tilde{\boldsymbol{\eta}})$$

Cada componente del error y su tangente hiperbólica tienen el mismo signo. Por tanto, $\dot{V}_k \leq 0$. La función no aumenta y el error converge bajo las hipótesis del modelo. Para la compensación dinámica se construye una función análoga con $\tilde{\boldsymbol{\nu}}$.

Esta demostración es útil, pero debe interpretarse con prudencia. Describe el modelo matemático y supone que las variables están disponibles, que el jacobiano es válido y que los actuadores cumplen las órdenes. La validación experimental sigue siendo indispensable.

7.5 Tres familias de controladores

El estudio comparativo de Guevara *et al.* (2023a) evalúa tres alternativas: cinemática inversa, compensación dinámica inversa y NMPC. La Figura 16 resume el compromiso general. La Tabla 8 compara las propiedades principales de las tres alternativas.

Figura 16.

Compromiso cualitativo entre precisión, consideración de restricciones y costo computacional

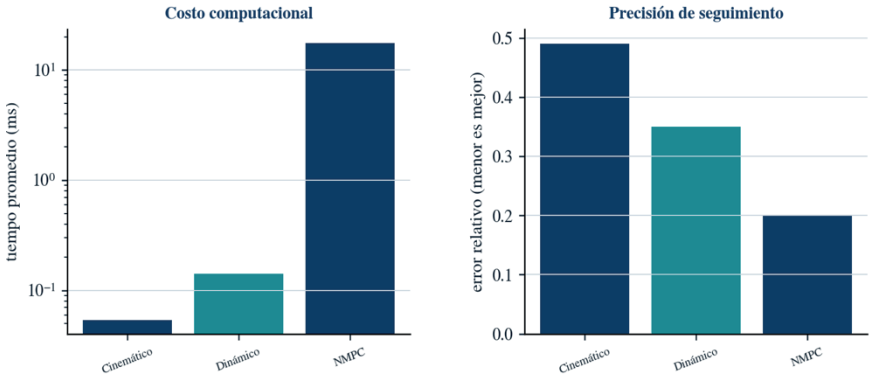


Tabla 8. *Comparación de controladores de realimentación y control predictivo*

Controlador	**Modelo utilizado**	**Restricciones explícitas**	**Costo de cálculo**	**Situación favorable**
Cinemático inverso	Jacobiano	No, solo saturación suave.	Muy bajo	Seguimiento suave y hardware limitado
Compensación dinámica	Cinemática + dinámica reducida	No, puede limitarse la orden.	Bajo	Dinámica relevante, pero sin obstáculos ni límites complejos
NMPC	Modelo discreto sobre un horizonte	Sí. Estados, entradas, variaciones y seguridad.	Medio o alto	Trayectorias rápidas, límites duros y tareas de seguridad

7.6 Comparación experimental

En simulación, los tiempos promedio reportados fueron 0.054 ms para el controlador cinemático, 0.141 ms para la compensación dinámica y 17.54 ms para NMPC. En vuelo real, aumentaron a 0.075 ms, 0.26 ms y 25.36 ms, respectivamente. Aun siendo mucho más costoso, el NMPC permaneció dentro del período de ejecución utilizado. La Figura 17 y la Tabla 9 presentan los tiempos de cálculo reportados.

Figura 17.

Tiempos promedio de cálculo reportados para los tres controladores

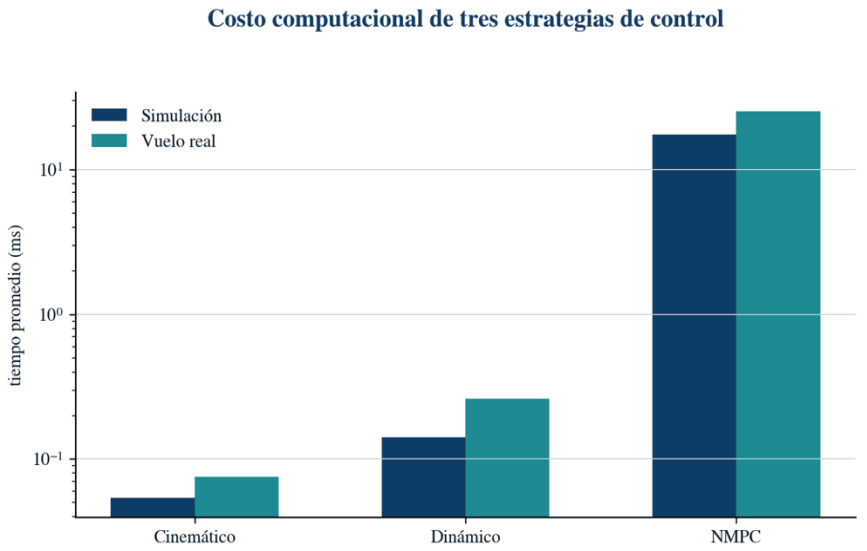


Tabla 9. *Tiempos promedio de cálculo reportados*

Entorno	Cinemático inverso (ms)	Compensación dinámica (ms)	NMPC (ms)
Simulación	0.054	0.141	17.54
Vuelo real	0.075	0.26	25.36

En los ensayos reales con una DJI Matrice 100, el NMPC presentó los menores límites aproximados de error: menos de 0.10 m en x, 0.15 m en y, y 0.20 m en z. La comparación se realizó con una trayectoria de Lissajous y bajo viento, cuya magnitud reportada alcanzó aproximadamente 21.5 km/h. El resultado no implica que el NMPC sea siempre obligatorio; muestra que su costo adicional puede justificarse cuando la trayectoria exige más precisión y robustez.

Idea clave. El tiempo de cálculo debe compararse con el período de muestreo, no con cero. Un controlador de 25 ms es viable a 30 Hz, si el proceso completo termina antes de aproximadamente 33 ms y conserva margen para comunicación y estimación.

7.7 Cómo seleccionar un controlador base

Un procedimiento práctico es comenzar por la estructura más simple que cumpla la misión:

1. Formular y validar el controlador cinemático.
2. Verificar si el error proviene de la dinámica de velocidad.
3. Añadir compensación dinámica, si la respuesta interna es relevante.
4. Migrar a NMPC cuando existan restricciones múltiples, obstáculos, referencias agresivas o una necesidad clara de anticipación.

Este orden evita utilizar optimización para corregir problemas que en realidad provienen de unidades, marcos de referencia o identificación deficiente.

7.8 Guía de implementación general con apoyo de IA

Los controladores de realimentación son la primera referencia práctica porque son fáciles de ejecutar, auditar y utilizar como respaldo de seguridad.

Paso 1. Implementar el cálculo del error respetando periodicidad angular y marcos de referencia. Un error de yaw no debe crecer artificialmente al atravesar $\pm\pi$.

Paso 2. Programar la ley cinemática y la compensación dinámica en módulos separados, con saturaciones explícitas y límites de variación de la orden.

Paso 3. Iniciar con ganancias pequeñas, aumentar un canal a la vez y documentar el efecto sobre tiempo de establecimiento, sobreimpulso y ruido.

Paso 4. Añadir una máquina de estados para armado, seguimiento, flotación, retorno y aborto. El controlador matemático no reemplaza la lógica operacional.

Paso 5. Solicitar a la IA pruebas unitarias, análisis dimensional y explicación de cada término. No aceptar código que oculte conversiones de unidades, signos o saturaciones.

7.9 Resumen del capítulo

Los controladores de realimentación ofrecen soluciones transparentes y de bajo costo. La cinemática inversa corrige la pose; la compensación

dinámica añade la respuesta de velocidad. Una función de Lyapunov permite razonar sobre convergencia. El NMPC mejora precisión y manejo de restricciones, pero requiere una plataforma computacional y una formulación más cuidadosa.

7.10 Preguntas de repaso

1. ¿Qué diferencia existe entre corregir el error de pose y corregir el error de velocidad?
2. ¿Qué función cumple la tangente hiperbólica en las leyes de control?
3. ¿Por qué una demostración de Lyapunov no reemplaza la validación experimental?
4. ¿Qué información adicional necesita la compensación dinámica?
5. ¿En qué escenario elegiría un controlador cinemático en lugar de un NMPC?

8. Control predictivo no lineal

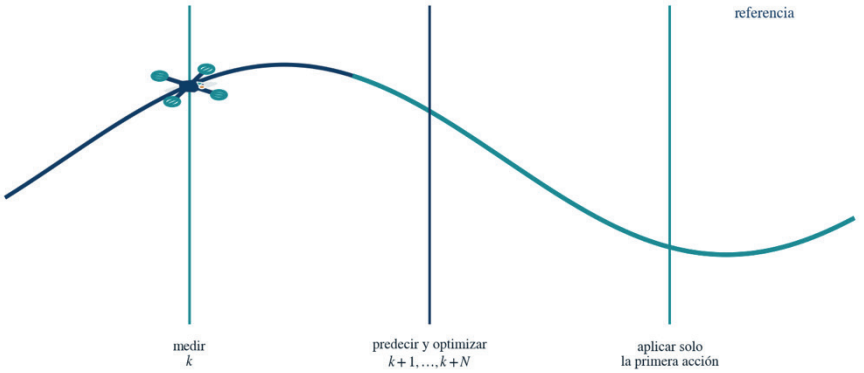
Objetivos del capítulo. Comprender el horizonte deslizante, formular una función de costo, incorporar restricciones dinámicas y geométricas, reconocer el papel de la discretización y el solver, y establecer un proceso de ajuste y verificación en tiempo real. Se distinguirán con claridad la formulación continua de diseño y la transcripción discreta que realmente resuelve el optimizador, así como las restricciones que cada misión declara.

8.1 Predecir antes de actuar

El control predictivo basado en modelo (Model Predictive Control, MPC) calcula una secuencia de acciones futuras. Solo aplica la primera y, al llegar una nueva medición, repite la optimización. Este mecanismo se conoce como horizonte deslizante (Figura 18).

Figura 18.

Horizonte deslizante: en cada instante se optimiza una secuencia y se aplica únicamente la primera acción



La anticipación permite reducir una velocidad antes de llegar a una curva, distribuir el esfuerzo de control y considerar un obstáculo antes de estar demasiado cerca. El precio es resolver repetidamente un problema de optimización.

8.2 Modelo de predicción

Sea

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}),$$

con estado $\mathbf{x}$ y entrada $\mathbf{u}$, después de discretizar con período Δt se obtiene:

$$\mathbf{x}_{k+1} = \mathbf{F}_d(\mathbf{x}_k, \mathbf{u}_k)$$

En los trabajos de esta obra, el estado puede incluir pose y velocidades de maniobrabilidad:

$$\mathbf{x} = [\boldsymbol{\eta}^T \quad \boldsymbol{\nu}^T]^T$$

La entrada es la referencia enviada al controlador interno del vehículo. Esta elección es importante: el modelo debe predecir la respuesta a la orden que realmente está disponible mediante el SDK.

Antes de discretizar conviene declarar el problema de horizonte finito en tiempo continuo. Esta forma especifica el modelo, el costo y las condiciones admisibles; las funciones de restricción contienen únicamente los límites que la aplicación ha definido y medido.

$$\min_{\mathbf{x}(\cdot),\,\boldsymbol{\mu}(\cdot)} J_c = \int_0^T \ell_c\left(\mathbf{x}(\tau), \boldsymbol{\mu}(\tau)\right) d\tau + V_f\big(\mathbf{x}(T)\big)$$

$$\mathbf{x}(0) = \hat{\mathbf{x}}_k, \qquad \dot{\mathbf{x}} = \mathbf{f}_c(\mathbf{x}, \boldsymbol{\mu})$$

$$\mathbf{h}(\mathbf{x}, \boldsymbol{\mu}) \leq \mathbf{0}, \qquad \mathbf{h}_f\big(\mathbf{x}(T)\big) \leq \mathbf{0}$$

La expresión corresponde a la forma general de control óptimo de horizonte finito; no presupone que todas las misiones utilicen las mismas desigualdades ni una región terminal idéntica (Rawlings *et al.*, 2020, Sección 8.5).

8.3 Función de costo

Después de elegir el período de muestreo y el integrador, la formulación discreta de seguimiento se escribe con estados y acciones predichos. La primera acción óptima es la única que se aplica antes de actualizar la medición.

$$\min_{\{\mathbf{x}_{i|k},\,\boldsymbol{\mu}_{i|k}\}} J_d = \sum_{i=0}^{N-1} \ell_d\left(\mathbf{x}_{i|k}, \boldsymbol{\mu}_{i|k}\right) + V_f\left(\mathbf{x}_{N|k}\right)$$

$$\mathbf{x}_{0|k} = \hat{\mathbf{x}}_k, \qquad \mathbf{x}_{i+1|k} = \mathbf{F}_d\left(\mathbf{x}_{i|k}, \boldsymbol{\mu}_{i|k}\right)$$

$$\mathbf{h}_d\left(\mathbf{x}_{i|k}, \boldsymbol{\mu}_{i|k}\right) \leq \mathbf{0}, \qquad \mathbf{h}_f\left(\mathbf{x}_{N|k}\right) \leq \mathbf{0}, \qquad i = 0, \ldots, N-1$$

La función de etapa reúne solamente los objetivos que la misión ha declarado para el horizonte, mientras que la función terminal expresa la condición al cierre de la predicción. Del mismo modo, las funciones de restricción de trayectoria y terminales representan límites concretos, no una lista predeterminada.

Aumentar un peso no "mejora" automáticamente esa variable. Si todos los pesos son grandes o están mal escalados, el problema puede quedar mal condicionado. Antes de ajustar, conviene normalizar variables en función de sus rangos físicos.

8.4 Restricciones

Una restricción no se agrega por ser usual, sino porque representa un límite verificable de la plataforma, de la misión o de la percepción disponible. Por ello, deben declararse la señal restringida, sus unidades, el instante de horizonte, el sentido de la desigualdad y la fuente de la medición.

$$\mathbf{x}_{\min} \leq \mathbf{x}_i \leq \mathbf{x}_{\max}$$

$$\mathbf{u}_{\min} \leq \mathbf{u}_i \leq \mathbf{u}_{\max}$$

$$\Delta\mathbf{u}_{\min} \leq \mathbf{u}_i - \mathbf{u}_{i-1} \leq \Delta\mathbf{u}_{\max}$$

Los casos de estudio no emplean un catálogo universal de restricciones: cada uno formaliza solamente las condiciones de su tarea. La Tabla 10 identifica las restricciones declaradas en los controladores

predictivos analizados y conserva la referencia que las respalda. Cuando una condición puede entrar en conflicto con otra, una holgura penalizada debe interpretarse como una decisión de diseño y reportarse explícitamente.

Tabla 10. *Restricciones declaradas en los controladores predictivos analizados*

Caso	Restricción declarada	Propósito en el problema	Referencia
Hexacóptero	Velocidad, variación de la orden y distancia a obstáculos	SDK y evitación de colisión	Recalde *et al.* (2022a)
NMPCC	Errores de contorno y avance; CLF y CBF	Progreso, convergencia y seguridad	Guevara *et al.* (2025)
Servo visual	Imagen, profundidad, velocidad y variación	Rasgos visibles y acción factible	Recalde *et al.* (2022b)

8.5 Evitación de obstáculos

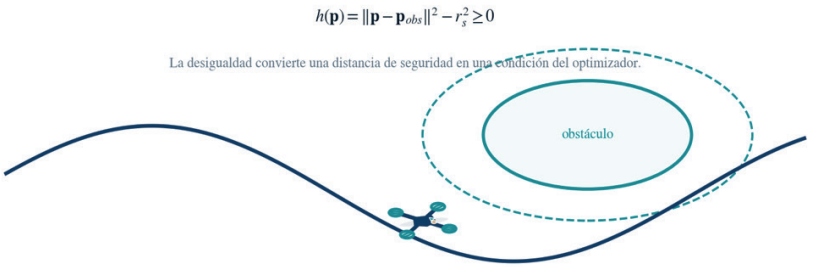
Un obstáculo esférico puede representarse mediante:

$$h(\mathbf{p}) = \| \mathbf{p} - \mathbf{p}_{obs} \|^2 - r_s^2 \geq 0,$$

con $r_s = r_{uav} + r_{obs} + margen$. El controlador debe mantener la posición fuera de la región prohibida. La Figura 19 representa dicha restricción geométrica.

Figura 19.

Restricción geométrica de seguridad alrededor de un obstáculo



En el hexacóptero estudiado por Recalde *et al.* (2022a), el problema incluye límites de velocidades, variación de acciones, modelo dinámico y obstáculos estáticos y móviles. La evitación se validó en simulación, mientras que el seguimiento de trayectoria, en vuelo real. La diferencia se debió a que la plataforma experimental no disponía del sensor requerido para medir obstáculos durante el vuelo.

8.6 Del problema continuo al programa no lineal

El problema de control óptimo debe transformarse en un conjunto finito de variables. En disparo simple, las acciones se optimizan y los estados se obtienen integrando desde el inicio. En disparo múltiple, los estados en varios nodos también son variables y se agregan restricciones de continuidad:

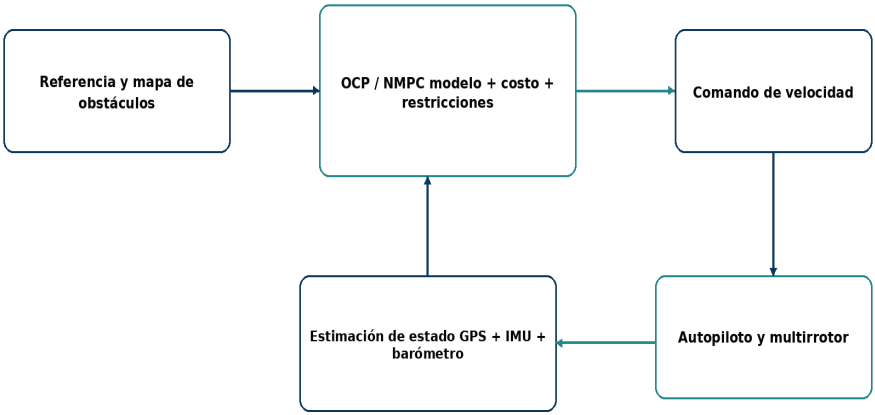
$$\mathbf{x}_{i+1} - \mathbf{F}_d(\mathbf{x}_i, \mathbf{u}_i) = \mathbf{0}$$

El disparo múltiple aumenta el número de variables, pero suele mejorar convergencia y manejo de problemas no lineales. Se utilizan

CasADi (Computer Algebra System with Automatic Differentiation) para diferenciación automática y formulación simbólica; según el caso, el problema se resuelve con Interior Point OPTimizer (IPOPT) o Acados. La Figura 20 resume la arquitectura general del NMPC. En términos de implementación, la transcripción transforma la dinámica y las condiciones de trayectoria en igualdades y desigualdades de un programa no lineal; no modifica el significado físico de las restricciones ni justifica añadir límites que la misión no haya especificado.

Figura 20.

Arquitectura general del NMPC: medición, predicción, optimización y aplicación de la primera acción



8.7 Tiempo real

La condición mínima es:

$$t_{estimación} + t_{optimización} + t_{comunicación} < \Delta t$$

No basta con reportar el promedio. Deben analizarse máximos, percentiles y picos durante maniobras difíciles. En el estudio del

hexacóptero, el período fue de 0.1 s y el tiempo de cálculo se mantuvo por debajo de 100 ms. Los picos crecieron al activar restricciones de obstáculos, pero permanecieron dentro del tiempo disponible.

Un horizonte largo mejora anticipación, aunque aumenta costo y sensibilidad al modelo. Un horizonte corto es más rápido, pero puede "ver" el obstáculo demasiado tarde. El ajuste debe considerar velocidad, radio de giro y distancia de frenado.

8.8 Proceso de ajuste

Un orden útil es:

1. Verificar el modelo sin controlador predictivo.
2. Resolver el NMPC sin restricciones complejas.
3. Ajustar **Q** y **R** con referencias suaves.
4. Añadir límites de entrada y variación.
5. Incorporar una restricción de seguridad a la vez.
6. Medir tiempo de cálculo y condición del solver.
7. Probar ruido, retardo, viento y errores iniciales.
8. Pasar de la validación Model-in-the-Loop (MIL) a HIL antes del vuelo real.

Advertencia. Cuando el solver falla, la causa no siempre es "falta de potencia". También puede existir un modelo discontinuo, variables mal escaladas, una condición inicial incompatible o restricciones mutuamente contradictorias.

8.9 Resultados que ilustran la utilidad del NMPC

El trabajo del hexacóptero identificó la dinámica mediante DMDc y utilizó ese modelo en el problema de control óptimo (OCP). En vuelo

real, el vehículo siguió la referencia bajo viento cercano a 10.1 km/h y los errores convergieron hacia valores próximos a cero. En simulación, evitó obstáculos estáticos y móviles respetando un radio de seguridad. El estudio comparativo del capítulo anterior mostró que el NMPC redujo el error frente a controladores de realimentación a cambio de decenas de milisegundos de cálculo.

Estos resultados enseñan dos principios. Primero, la capacidad de imponer restricciones es una propiedad estructural del método, no un parche posterior. Segundo, la precisión depende del modelo y de la información disponible: el optimizador no puede anticipar un obstáculo que ningún sensor ni mapa le comunica.

8.10 Guía de implementación general con apoyo de IA

Un NMPC replicable se construye en capas. Primero debe resolver un problema sin obstáculos; después se añaden límites, estabilidad y seguridad.

Paso 1. Definir estado, entrada, período de muestreo y horizonte en un archivo de configuración. Las escalas de las variables deben ser comparables o normalizarse de forma explícita.

Paso 2. Implementar la dinámica discreta y verificarla fuera del optimizador. La misma función debe reproducir la evolución utilizada por el solver.

Paso 3. Construir el costo con términos identificables y registrar cada peso. Cambiar un peso por vez y conservar las configuraciones que se comparan.

Paso 4. Añadir restricciones de entrada, variación, estado y obstáculos de forma incremental. Cuando el problema sea inviable,

informar qué restricción se viola y no reutilizar indefinidamente una orden antigua.

Paso 5. Medir tiempo de preparación, solución y comunicación por separado. La IA puede traducir la formulación a CasADi, acados, MATLAB (Matrix Laboratory), Julia u otra biblioteca, pero debe conservarse una prueba numérica independiente del solver.

Paso 6. Implementar calentamiento con la solución anterior, política de respaldo y registro del estado de convergencia antes de avanzar a HIL o vuelo.

8.11 Resumen del capítulo

El NMPC resuelve repetidamente un problema de control óptimo sobre un horizonte. Su función de costo equilibra seguimiento y esfuerzo, mientras las restricciones representan límites físicos y seguridad. La discretización, el escalado, el horizonte y el solver determinan la viabilidad en tiempo real.

8.12 Preguntas de repaso

1. ¿Por qué se aplica solo la primera acción de la secuencia óptima?
2. ¿Qué diferencia existe entre una restricción dura y una blanda?
3. ¿Qué ventaja ofrece el disparo múltiple?
4. ¿Por qué el tiempo máximo de cálculo es más informativo que el promedio?
5. ¿Qué ocurre si el horizonte es más corto que el tiempo necesario para evitar un obstáculo?

9. Dinámica no modelada, observadores y adaptación

Objetivos del capítulo. Identificar fuentes de dinámica no modelada, representar su efecto como una perturbación equivalente, comprender el papel de una ley adaptativa y de un observador de momento, y analizar por qué su combinación mejora robustez sin sustituir la necesidad de un modelo nominal. La compensación añadida se evaluará de manera aislada antes de integrarla, para identificar si la mejora proviene de la adaptación, del observador o de su combinación.

9.1 Errores en los modelos

Un modelo reducido omite deliberadamente detalles. En un cuadricóptero comercial, por ejemplo, puede asumir que *roll* y *pitch* son pequeños y que los controladores internos convierten inmediatamente una referencia en velocidad. Esa aproximación facilita el control, pero no incluye todos los acoplamientos, efectos aerodinámicos, variaciones de masa o retardos.

Es útil agrupar la diferencia entre el modelo nominal y la planta real como una perturbación equivalente:

$$\boldsymbol{\mu}_T = \mathbf{M}_n\dot{\boldsymbol{\nu}} + \mathbf{C}_n(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{\tau}_u$$

El término $\boldsymbol{\tau}_u$ no tiene que corresponder a una sola fuerza física. Representa aquello que el modelo nominal no explica en el espacio de las entradas de velocidad. La Tabla 11 compara las estrategias frente a la dinámica no modelada.

9.2 Cuatro respuestas posibles

Tabla 11. *Estrategias frente a dinámica no modelada*

Estrategia	Idea	Ventaja	Limitación
Aumentar el modelo físico	Añadir más fenómenos y parámetros.	Interpretación detallada	Mayor complejidad e identificación difícil.
Identificar en línea	Actualizar **A** y **B**.	Sigue cambios persistentes	Necesita excitación informativa.
Ley adaptativa	Estimar parámetros o una entrada equivalente.	Compensación de incertidumbre estructurada	Hay convergencia dependiente de las señales.
Observador de perturbación	Reconstruir el efecto externo a partir del error dinámico.	Respuesta rápida	Es sensible a ruido, derivadas y ajuste de ganancias.

La investigación de Guevara *et al.* (2024) combina las dos últimas opciones con NMPC. El resultado se denomina NMPC adaptativo con observador dinámico de momento.

9.3 NMPC nominal

El controlador nominal utiliza

$$\dot{\mathbf{x}} = \mathbf{f}_n(\mathbf{x}, \boldsymbol{\mu}),$$

sin conocer $\boldsymbol{\tau}_u$. Si la perturbación es pequeña, la realimentación corrige el error. Si es grande o persistente, la predicción se sesga y el vehículo puede mantener una desviación.

El estudio usa una función de costo cuadrática para posición y entrada, un horizonte de un segundo con $N = 30$ y una tasa de 30 Hz. En el entorno MIL, el tiempo reportado para resolver el problema de

control óptimo (OCP) se mantuvo por debajo de 5.2 ms, aunque este valor depende del hardware y de la formulación concreta.

9.4 Ley adaptativa

La dinámica desconocida puede aproximarse con un regresor:

$$\boldsymbol{\tau}_u \approx \mathbf{Y}_u(\dot{\boldsymbol{\nu}}, \boldsymbol{\nu})\boldsymbol{\xi}_u$$

La estimación utiliza $\hat{\boldsymbol{\xi}}_u$ y genera una compensación

$$\hat{\boldsymbol{\mu}}_u = \mathbf{Y}_u\hat{\boldsymbol{\xi}}_u$$

Una ley adaptativa típica modifica los parámetros en la dirección que reduce el error:

$$\dot{\hat{\boldsymbol{\xi}}}_u = -\boldsymbol{\Gamma}\mathbf{Y}_u^T s_f$$

con $\boldsymbol{\Gamma} > 0$ y una señal de error filtrada $\mathbf{s}$. El objetivo no es identificar cada parámetro físico por separado, sino generar una acción equivalente que reduzca la discrepancia.

9.5 Observador dinámico de momento

El momento generalizado puede definirse como:

$$\mathbf{p} = \mathbf{M}_n\boldsymbol{\nu}$$

Al comparar su evolución esperada con la observada se obtiene un residual relacionado con fuerzas o velocidades externas. Un observador de primer orden puede escribirse conceptualmente como:

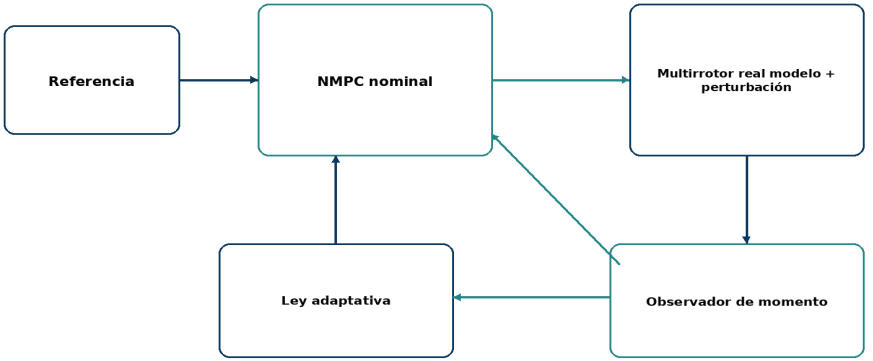
$$\dot{\hat{\boldsymbol{\tau}}}_{ext} = \mathbf{K}_o(\mathbf{r} - \hat{\boldsymbol{\tau}}_{ext})$$

En la fórmula, $\mathbf{r}$ se construye a partir de las velocidades medidas, la dinámica nominal y la entrada ordenada. Una ganancia grande responde con rapidez, pero amplifica el ruido; una ganancia pequeña

filtra mejor, aunque introduce retardo. La Figura 21 presenta la arquitectura del NMPC adaptativo con observador.

Figura 21.

Arquitectura del NMPC adaptativo con observador dinámico de momento



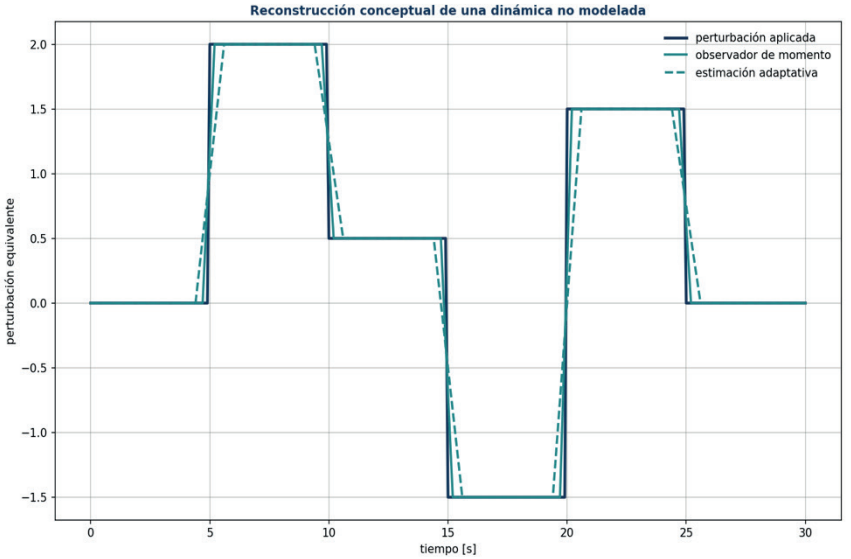
9.6 Combinación de adaptación y observación

La ley adaptativa aprende una compensación sostenida, mientras el observador detecta cambios rápidos. Al combinarlos, la entrada puede organizarse como:

$$\boldsymbol{\mu} = \boldsymbol{\mu}_{NMPC} + \hat{\boldsymbol{\mu}}_u - \hat{\boldsymbol{\mu}}_{ext}$$

Figura 22.

Reconstrucción conceptual de la perturbación mediante modelo nominal, observador y adaptación



Los signos exactos dependen de la convención, pero la idea es clara: el NMPC realiza el seguimiento, la adaptación aproxima la dinámica faltante y el observador reconstruye la perturbación externa o el error residual. La Figura 22 resume la reconstrucción conceptual de la perturbación.

Una combinación redundante puede parecer innecesaria. Sin embargo, los dos mecanismos trabajan en escalas temporales distintas. El observador no necesita esperar convergencia paramétrica, y la adaptación puede corregir el sesgo que un observador filtrado deja en régimen permanente.

9.7 Interpretación de los resultados

La evaluación se realizó sobre un gemelo digital de una DJI Matrice 100 en Webots. Se aplicaron perturbaciones por tramos y señales sinusoidales en los cuatro componentes de velocidad. El NMPC nominal mostró errores apreciables. La adaptación redujo el error y aproximó las perturbaciones; el observador mejoró la reconstrucción y la combinación mantuvo el error final característico por debajo de 0.1 en la norma utilizada por el estudio.

El resultado debe leerse como una validación MIL de robustez, no como una demostración universal. La eficacia depende de calidad de medición, frecuencia, parámetros del observador y persistencia de excitación. Antes de transferir al vuelo real deben analizarse ruido de IMU, cuantización, retardos y límites de acción.

9.8 Estabilidad, acotación y robustez

En sistemas adaptativos es común demostrar que una función compuesta,

$$V = V_{seguimiento} + \frac{1}{2}\tilde{\boldsymbol{\xi}}_u^T \boldsymbol{\Gamma}^{-1} \tilde{\boldsymbol{\xi}}_u,$$

no crece. Esto garantiza acotación de errores bajo determinadas hipótesis. No siempre garantiza que los parámetros estimados sean idénticos a los físicos. Para ello se requiere excitación persistente. Desde la perspectiva de control, una compensación correcta puede ser suficiente, aunque el parámetro carezca de una interpretación física exacta.

Advertencia. Diferenciar señales medidas para construir un observador puede amplificar ruido. Es preferible utilizar formulaciones

integrales, filtros, estimadores de estado o derivadas obtenidas dentro del modelo cuando sea posible.

9.9 Proceso recomendado

1. Cuantificar el error del modelo nominal.
2. Determinar si la discrepancia es lenta, rápida o ambas.
3. Diseñar el observador con perturbaciones simuladas conocidas.
4. Añadir límites a la compensación estimada.
5. Activar la adaptación con una ganancia conservadora.
6. Verificar que cada bloque, por separado, mejora el sistema.
7. Evaluar la combinación y medir la sensibilidad al ruido.
8. Implementar un modo seguro que regrese al controlador nominal si la estimación deja de ser plausible.

9.10 Guía de implementación general con apoyo de IA

La estimación de dinámica no modelada se incorpora únicamente después de validar el controlador nominal. De otro modo, el observador puede ocultar errores de signos, marcos o parámetros.

Paso 1. Definir la perturbación equivalente y el punto exacto donde se suma al modelo. Evitar la mezcla de perturbaciones expresadas en fuerza con correcciones expresadas en velocidad.

Paso 2. Implementar el observador con filtros y condiciones iniciales conocidas. Registrar por separado la innovación, la estimación y la compensación aplicada.

Paso 3. Limitar la amplitud y velocidad de adaptación; incluir proyección o saturación para impedir que una medición atípica modifique de forma permanente los parámetros.

Paso 4. Ensayar perturbaciones reproducibles: cambio de masa, viento simulado, sesgo y retraso. La IA puede generar escenarios y scripts de análisis, pero la señal inyectada debe quedar documentada.

Paso 5. Comparar cuatro casos con el mismo protocolo: nominal, adaptativo, observador y combinación. Reportar error, esfuerzo de control, tiempo de cómputo y estabilidad de la estimación.

9.11 Resumen del capítulo

La dinámica no modelada puede representarse como una perturbación equivalente. La adaptación corrige discrepancias estructuradas y el observador de momento reconstruye cambios externos. Combinados con NMPC, mejoran robustez, pero añaden estados, ganancias y riesgos que deben validarse progresivamente.

9.12 Preguntas de repaso

1. ¿Qué fenómenos puede agrupar el término de *dinámica no modelada*?
2. ¿Qué diferencia conceptual existe entre *adaptación* y *observación*?
3. ¿Por qué la convergencia del error no implica identificación exacta de parámetros?
4. ¿Qué efecto tiene una ganancia de observador demasiado alta?
5. ¿Qué mecanismo de seguridad debería acompañar a una compensación adaptativa?

10. Seguimiento seguro de caminos

Objetivos del capítulo. Diferenciar seguimiento de camino y de trayectoria, formular errores de contorno y avance, representar orientación con cuaterniones, comprender funciones de Lyapunov y de barrera y describir su integración en un control predictivo de contorno. La seguridad se verificará con métricas geométricas y de factibilidad, porque una trayectoria visualmente aceptable no prueba por sí sola el cumplimiento de una barrera.

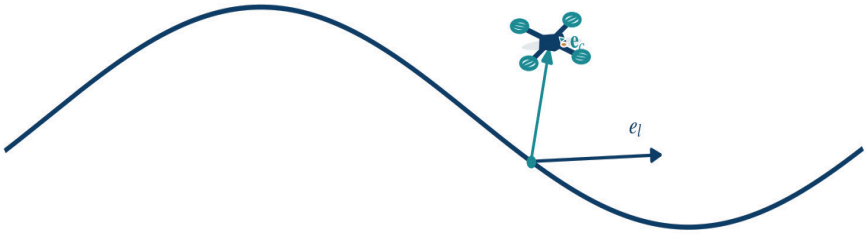
10.1 Del seguimiento temporal al seguimiento geométrico

Una trayectoria impone una posición para cada instante. Un camino define una curva, pero permite decidir la velocidad de avance. Esta libertad es valiosa cuando el vehículo debe desacelerar ante un obstáculo o una zona difícil.

Sea un camino $\mathbf{p}_d(\vartheta)$ parametrizado por ϑ, el controlador no solo calcula las acciones del vehículo aéreo no tripulado (Unmanned Aerial Vehicle, UAV), sino también optimiza una velocidad de progreso $v_\vartheta = \dot{\vartheta}$. La Figura 23 descompone el error en contorno y avance.

Figura 23.

Descomposición del error en componente de contorno y componente de avance



10.2 Error de contorno y error de avance

En torno a un punto del camino se define un vector tangente unitario $\mathbf{t}(\vartheta)$. El error de posición es:

$$\mathbf{e} = \mathbf{p} - \mathbf{p}_d(\vartheta)$$

El error de avance o *lag* es la proyección tangencial:

$$\mathbf{e}_l = \mathbf{t}\mathbf{t}^T\mathbf{e}$$

El error de contorno es la componente ortogonal:

$$\mathbf{e}_c = (\mathbf{I} - \mathbf{t}\mathbf{t}^T)\mathbf{e}$$

Penalizar $\mathbf{e}_c$ mantiene al vehículo cerca de la geometría. Penalizar $\mathbf{e}_l$ evita que el parámetro de avance se aleje de la posición real. Un término adicional premia v_ϑ para completar el camino con rapidez.

10.3 Parametrización por longitud de arco

La longitud recorrida desde t_0 hasta t es:

$$\vartheta(t) = \int_{t_0}^{t} \left\| \frac{d\mathbf{p}_d(\tau)}{d\tau} \right\| d\tau$$

Cuando no existe una inversa analítica $t(\vartheta)$, puede utilizarse bisección. Se propone un intervalo temporal, se evalúa la longitud y se reduce el intervalo hasta encontrar el tiempo correspondiente. Esta parametrización hace que v_ϑ tenga una interpretación física de velocidad sobre el camino.

10.4 Error de orientación con cuaterniones

La orientación deseada $\mathbf{q}_d$ y la real $\mathbf{q}$ generan el cuaternión de error

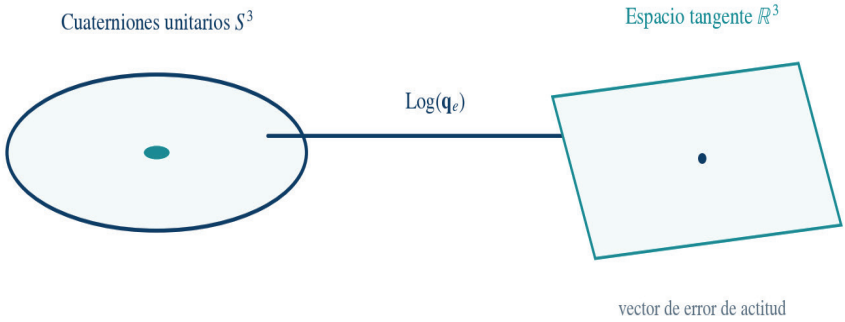
$$\tilde{\mathbf{q}} = \mathbf{q}_d \otimes \mathbf{q}^{-1}$$

Para evitar la doble representación, se elige el signo que mantenga positiva la parte escalar. Después, el mapa logarítmico lleva el error a $\mathbb{R}^3$:

$$\mathrm{Log}(\tilde{\mathbf{q}}) = 2\,\tilde{\mathbf{q}}_v \frac{\mathrm{atan2}(\| \tilde{\mathbf{q}}_v \|, \tilde{q}_w)}{\| \tilde{\mathbf{q}}_v \|}$$

Figura 24.

El mapa Log convierte el error de cuaternión en un vector del espacio tangente



Esta operación permite incluir la orientación en una función de costo cuadrática sin utilizar directamente cuatro componentes redundantes. La Figura 24 ilustra esta transformación al espacio tangente.

10.5 Función de Lyapunov de control

Para un sistema afín

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u},$$

una función de Lyapunov de control (Control Lyapunov Function, CLF) $V(\mathbf{x})$ mide distancia a un objetivo. Una condición de estabilización exponencial es:

$$\dot{V}(\mathbf{x}, \mathbf{u}) \leq -\gamma V(\mathbf{x}), \qquad \gamma > 0$$

En tiempo discreto puede aproximarse como:

$$\frac{V(\mathbf{x}_{k+1}) - V(\mathbf{x}_k)}{\Delta t} \leq -\gamma V(\mathbf{x}_k) + \zeta,$$

con holgura $\zeta \geq 0$. La holgura evita que una exigencia rígida de estabilidad vuelva inviable el problema durante una maniobra de seguridad.

10.6 Función de barrera de control

Una región segura se define por

$$\mathcal{C} = \{\mathbf{x}: h(\mathbf{x}) \geq 0\}$$

Para un obstáculo,

$$h(\mathbf{p}) = \| \mathbf{p} - \mathbf{p}_{obs} \| - (r_{uav} + r_{obs} + margen)$$

La función es positiva fuera de la zona de riesgo, cero en la frontera y negativa dentro de ella. Una función de barrera de control (Control Barrier Function, CBF) impone una condición sobre h y sus derivadas para que el sistema permanezca en $\mathcal{C}$. La Figura 25 contrasta el papel de la CLF y la CBF, mientras que la Tabla 12 resume sus diferencias.

Cuando la entrada no aparece en la primera derivada de h, se utiliza una CBF de orden superior. En forma conceptual,

$$\ddot{h} + k_1 \dot{h} + k_0 h \geq 0$$

Figura 25.

La CLF impulsa el estado hacia el objetivo y la CBF impide abandonar el conjunto seguro

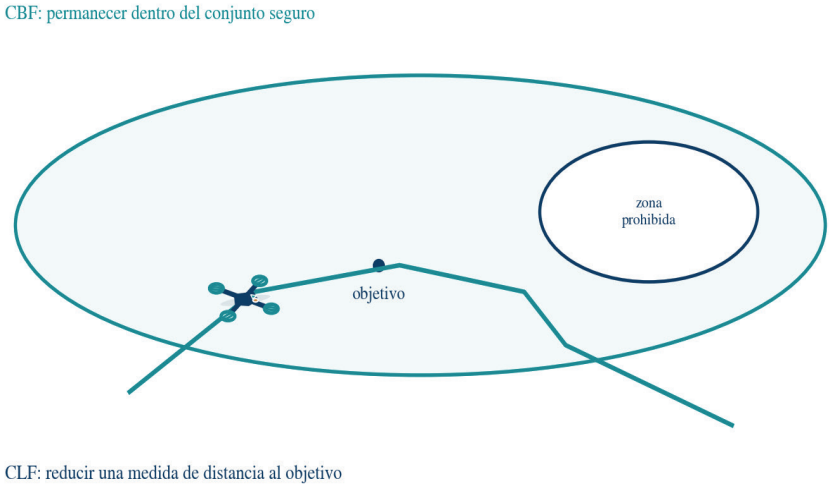


Tabla 12. *Diferencia entre función de Lyapunov y función de barrera*

Herramienta	**Pregunta que responde**	**Condición típica**	**Papel en el OCP**
CLF	¿El error converge?	$\dot{V} \leq -\gamma V$	Restricción o penalización de estabilidad
CBF	¿El estado permanece seguro?	$\dot{h} + \alpha(h) \geq 0$	Restricción de invariancia
CBF de orden superior	¿La entrada afecta a h después de varias derivadas?	$\ddot{h} + k_1\dot{h} + k_0 h \geq 0$	Seguridad para dinámica con grado relativo mayor que uno

10.7 Formulación del control predictivo no lineal de contorno (NMPCC)

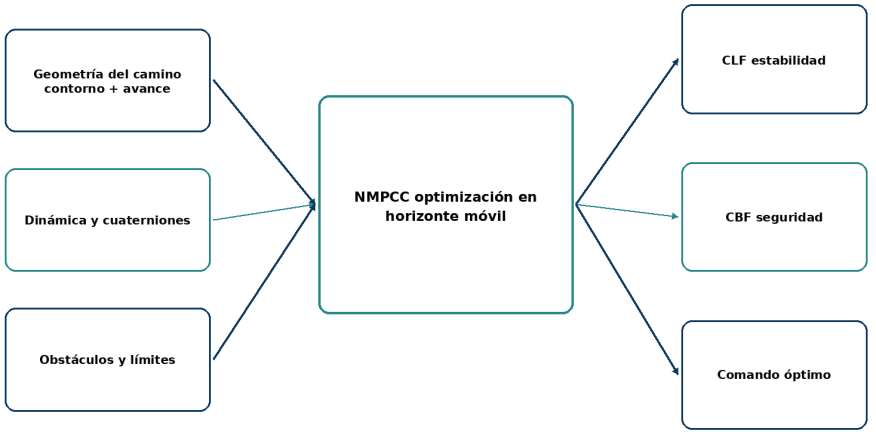
El control predictivo de contorno utiliza una función de costo como:

$$J = \sum_{k=0}^{N-1} \left(\| \mathbf{e}_{c,k} \|^2_{\mathbf{Q}_c} + \| \mathbf{e}_{l,k} \|^2_{\mathbf{Q}_l} + \| \mathrm{Log}(\tilde{\mathbf{q}}_k) \|^2_{\mathbf{Q}_q} + \| \mathbf{u}_k \|^2_{\mathbf{R}} - \mu v_{\vartheta,k} \right)$$

El signo negativo premia el avance. Las restricciones incluyen dinámica, límites de entrada, $0 \leq v_\vartheta \leq v_{\vartheta,max}$, CLF y CBF. La Figura 26 presenta la arquitectura integrada del NMPCC.

Figura 26.

Arquitectura integrada del NMPCC con errores geométricos, CLF, CBF y control de actitud



10.8 Validación experimental

Guevara *et al.* (2025) implementaron el método en una DJI Matrice 100 con una NVIDIA Jetson Orin NX. CasADi y Acados resolvieron

un horizonte de 30 pasos a 30 Hz, equivalente a un segundo. El vehículo siguió un camino en forma de ocho parametrizado por longitud de arco, con velocidad máxima de progreso de 5 m/s.

El experimento utilizó obstáculos virtuales con posiciones conocidas. Uno de ellos se desplazó en sentido contrario al UAV, aumentando la velocidad relativa. Durante la evitación, el error de contorno fue el más afectado, como es esperable: el vehículo se separa temporalmente del camino para mantener seguridad. Los valores instantáneos reportados permanecieron por debajo de 1.97 m para contorno y 1.87 m para avance en presencia de obstáculos.

La CBF se mantuvo positiva y la CLF forzó una tendencia de estabilidad. No obstante, esos resultados deben interpretarse junto con el tamaño del vehículo, el margen y la ubicación conocida de los obstáculos. La seguridad real exige además detección, incertidumbre de localización y un procedimiento ante pérdida de comunicación.

10.9 Conflicto entre seguimiento y seguridad

Seguir exactamente el camino y evitar un obstáculo pueden ser objetivos incompatibles. Una jerarquía razonable es:

1. Mantener la seguridad.
2. Respetar límites físicos.
3. Conservar estabilidad.
4. Reducir el error de contorno.
5. Maximizar progreso.

Las holguras y pesos deben reflejar esa jerarquía. Penalizar demasiado el contorno puede obligar al controlador a acercarse de forma peligrosa al obstáculo. Penalizar demasiado la barrera puede producir

desvíos innecesarios o bloqueo. La sintonía debe apoyarse en escenarios límite, no únicamente en un vuelo nominal.

10.10 Guía de implementación general con apoyo de IA

El seguimiento seguro exige distinguir entre el objetivo de desempeño y la condición que nunca debe violarse. La CLF y la CBF se prueban antes de integrarlas en el NMPCC completo.

Paso 1. Implementar la proyección sobre el camino y comprobar que los errores de contorno y avance cambian de forma continua cerca de cada segmento.

Paso 2. Verificar el error de orientación con cuaterniones y normalizar el cuaternión después de cada integración.

Paso 3. Evaluar la CLF sobre estados perturbados y confirmar que la condición de disminución es alcanzable con las entradas disponibles.

Paso 4. Evaluar la CBF en puntos interiores, fronterizos y exteriores al conjunto seguro. El signo de la función debe coincidir con la convención adoptada.

Paso 5. Integrar CLF y CBF con variables de holgura penalizadas y registrar cuándo se utilizan. La IA puede proponer expresiones simbólicas y derivadas, pero deben verificarse con diferenciación numérica y casos geométricos simples.

Paso 6. Repetir ensayos con posiciones de obstáculos conocidas y perturbadas; reportar distancia mínima, holgura, error de camino y tiempo de solución.

10.11 Resumen del capítulo

El seguimiento de caminos separa geometría y tiempo. El NMPCC minimiza errores de contorno y avance mientras optimiza el progreso. Los cuaterniones ofrecen una representación robusta de actitud; las CLF imponen convergencia y las CBF mantienen seguridad. Su integración permite formular una tarea compleja dentro de un único OCP.

10.12 Preguntas de repaso

1. ¿Qué libertad adicional ofrece un seguidor de camino frente a un seguidor de trayectoria?
2. ¿Qué representa el error de contorno?
3. ¿Por qué se utiliza el mapa Log para el error de orientación?
4. ¿Qué diferencia existe entre CLF y CBF?
5. ¿Por qué puede ser necesaria una CBF de orden superior?

Capítulo IV

Control de vuelo y seguridad




Z
X

11. Servo visual predictivo

Objetivos del capítulo. Comprender la proyección de puntos en una cámara, formular el error de servo visual basado en imagen, construir la matriz de interacción, incorporar la dinámica del multirrotor y mantener las características dentro del campo de visión mediante control predictivo. La cámara se tratará como parte de la cadena de control: su calibración, profundidad y campo de visión condicionan directamente las acciones factibles.

11.1 La cámara como sensor de realimentación

Un servo visual utiliza información de imagen para controlar el movimiento. En lugar de ordenar únicamente una posición global, puede buscar que un conjunto de puntos, líneas o regiones adopte una configuración deseada en el plano de la cámara. Esto resulta útil en interiores, inspección, aterrizaje, seguimiento de objetivos y aproximación a estructuras.

Las principales modalidades son el servo visual basado en imagen (Image-Based Visual Servoing, IBVS), el servo visual basado en posición (Position-Based Visual Servoing, PBVS) y los enfoques híbridos: la Tabla 13 compara estas modalidades.

Tabla 13. *Modalidades de servo visual*

Modalidad	Error utilizado	Ventaja	Limitación
IBVS	Coordenadas en la imagen	Menor dependencia de reconstrucción 3D	Trayectorias cartesianas indirectas y sensibilidad a profundidad
PBVS	Pose relativa 3D	Movimiento interpretable en el espacio	Dependencia de estimación de pose y calibración
Híbrido	Combinación de imagen y pose	Equilibrio de ambas propiedades	Mayor complejidad de diseño

11.2 Modelo de cámara

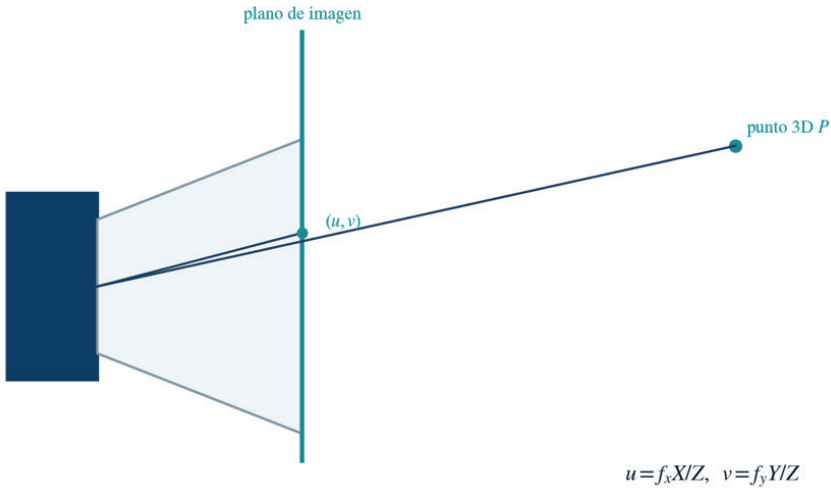
Para un punto $\mathbf{P}^C = [X, Y, Z]^T$ en el marco de la cámara, el modelo pinhole produce

$$u = f_x \frac{X}{Z} + u_0, \qquad v = f_y \frac{Y}{Z} + v_0,$$

donde (u_0, v_0) es el punto principal y $\left(f_x, f_y\right)$ son las distancias focales expresadas en píxeles. La profundidad Z debe ser positiva y su error influye en el control. La Figura 27 representa la proyección perspectiva.

Figura 27.

Proyección perspectiva de un punto tridimensional sobre el plano de la imagen



Para r características se construye:

$$\mathbf{s}_v = [u_1, v_1, \ldots, u_r, v_r]^T, \qquad \tilde{\mathbf{s}}_v = \mathbf{s}_{v,d} - \mathbf{s}_v$$

11.3 Matriz de interacción

La velocidad de una característica se relaciona con la velocidad de la cámara:

$$\dot{\mathbf{s}}_v = \mathbf{L}_s(\mathbf{s}_v, Z)\mathbf{v}_c$$

$$\dot{\mathbf{s}}_v = \mathbf{J}_v(\mathbf{s}_v, Z, \psi)\mathbf{v}$$

$$\mathbf{L}_p = \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \end{bmatrix}$$

Las matrices de todos los puntos se apilan. Después se transforma la velocidad del cuerpo del UAV a la cámara mediante la extrínseca. El modelo visual completo adopta la forma

$$\dot{\mathbf{s}} = \mathbf{J}_v(\mathbf{s}, \mathbf{Z}, \psi)\boldsymbol{\nu}$$

11.4 Control IBVS clásico

Una ley básica es:

$$\boldsymbol{\nu}_{ref} = \mathbf{J}_v^{\dagger}\mathbf{K}_2 \tanh(\mathbf{K}_2^{-1}\mathbf{K}_1\tilde{\mathbf{s}}_v)$$

Si todas las características permanecen visibles y la profundidad es razonable, el error tiende a cero. El problema surge cuando el estado inicial está lejos del deseado. La pseudoinversa puede ordenar un movimiento que expulse una característica del campo de visión. Una vez perdida, el controlador ya no dispone de la misma dimensión de error y puede cancelarse.

11.5 Modelo aumentado con dinámica del cuadricóptero

La cinemática de imagen describe cómo cambia la imagen para una velocidad dada. La plataforma comercial, sin embargo, no alcanza esa velocidad instantáneamente. Se identifica la dinámica reducida mediante DMDc:

$$\dot{\boldsymbol{\nu}} = \mathbf{A}\boldsymbol{\nu} + \mathbf{B}\boldsymbol{\nu}_{ref}$$

El estado aumentado es:

$$\boldsymbol{\chi} = [\mathbf{s}^T \quad \boldsymbol{\nu}^T]^T$$

y su evolución no lineal combina la cinemática de imagen, la transformación entre cámara y UAV y la dinámica de velocidad identificada. De esta forma, el optimizador no supone un actuador ideal e instantáneo.

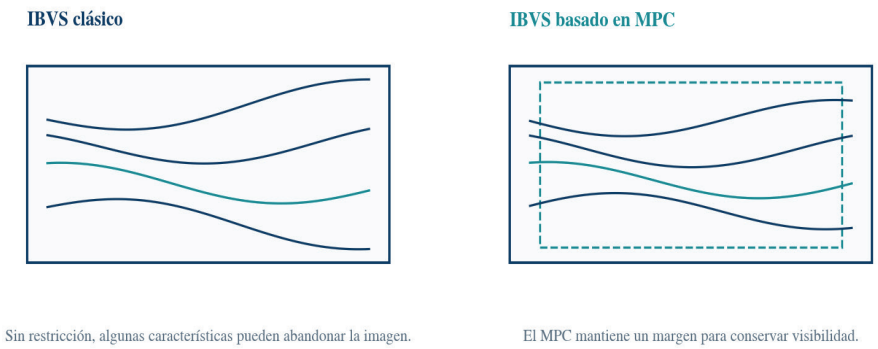
11.6 Restricciones del campo de visión (Field of View, FOV)

Para una imagen cuyos límites se definen alrededor del punto principal, cada característica debe satisfacer:

$$u_{min} \leq u_i \leq u_{max}, \qquad v_{min} \leq v_i \leq v_{max}$$

Figura 28.

Comparación conceptual entre un controlador que no impone visibilidad y un MPC que mantiene margen en el campo de visión



No conviene utilizar exactamente el borde como límite operativo. Una zona interior permite absorber ruido, retardo y error de calibración. También pueden añadirse restricciones de profundidad, velocidad y variación de la orden. La Figura 28 compara un controlador sin restricción de visibilidad con uno que mantiene margen en el campo de visión.

11.7 Formulación predictiva

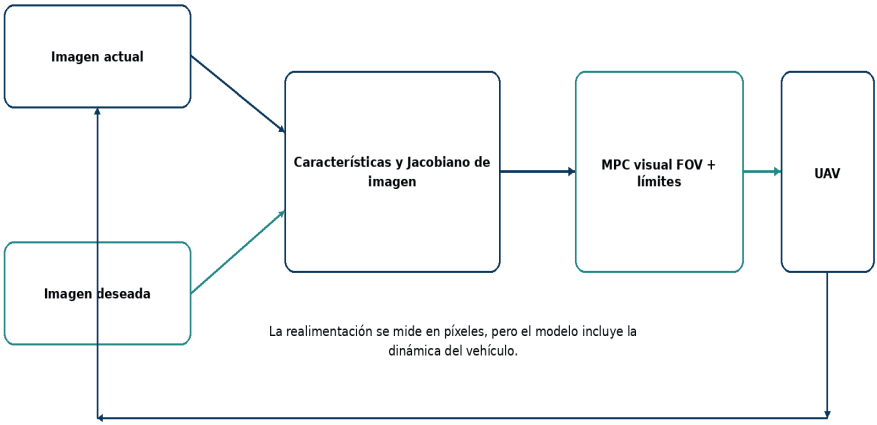
La función de costo visual puede escribirse como:

$$J = \sum_{k=0}^{N-1} \left(\| \tilde{\mathbf{s}}_{v,k} \|^2_{\mathbf{Q}_s} + \| \boldsymbol{\nu}_{ref,k} \|^2_{\mathbf{R}} \right) + \| \tilde{\mathbf{s}}_{v,N} \|^2_{\mathbf{Q}_N}$$

sujeta a la dinámica aumentada, los límites de imagen y los límites de entrada. En el experimento reportado, $\Delta t = 0.1$ s y $N = 10$, se produce nuevamente un horizonte de un segundo. La Figura 29 resume la arquitectura del servo visual predictivo.

Figura 29.

Arquitectura del servo visual predictivo: imagen, detección de rasgos, modelo aumentado, optimización y UAV



11.8 Entorno de simulación

La validación integra Webots, Open Source Computer Vision Library (OpenCV) y CasADi. Webots simula el Mavic Pro 2 y su cámara; OpenCV procesa la imagen y extrae las características; y el bloque de optimización resuelve el OCP restringido y envía referencias de velocidad. Esta separación facilita depuración: puede comprobarse por separado la calibración, la detección, la matriz de interacción, la dinámica y el solver.

El controlador clásico perdió dos de las cuatro características durante el ensayo y su ley se canceló. El MPC mantuvo todas las características dentro de la imagen y llevó el error hacia cero. El resultado muestra que la ventaja no proviene únicamente de "predecir mejor", sino de representar explícitamente la visibilidad como requisito.

11.9 Profundidad, calibración y robustez

El IBVS depende de Z. Puede medirse con cámara de color y profundidad (RGB-D), estimarse a partir de pose o aproximarse. Un error moderado suele afectar la velocidad de convergencia; uno grande puede producir movimientos inesperados. Además, deben verificarse:

- Convención de ejes de cámara y vehículo
- Unidades de focal y coordenadas
- Distorsión óptica
- Sincronización entre imagen y estado
- Pérdida temporal o identificación incorrecta de rasgos
- Movimiento del objetivo

Advertencia. Una característica dentro de la imagen no es necesariamente confiable. Puede estar ocluida, desenfocada o asociada al objeto equivocado. La restricción del campo de visión (Field of View, FOV) debe complementarse con una medida de calidad y una estrategia ante pérdida de detección.

11.10 Proceso de implementación

1. Calibrar intrínsecos y extrínsecos.
2. Verificar la proyección con puntos conocidos.
3. Implementar el IBVS clásico a baja velocidad.

4. Identificar la dinámica de velocidad.
5. Construir y discretizar el modelo aumentado.
6. Añadir límites de entrada.
7. Añadir restricciones de FOV con margen.
8. Probar errores iniciales grandes, oclusiones y profundidad incierta.
9. Pasar a HIL o vuelo controlado cuando el sistema pueda detenerse de forma segura ante pérdida de rasgos.

11.11 Guía de implementación general con apoyo de IA

El servo visual se implementa como una cadena de percepción y control. Cada etapa debe exponer su confianza y su marca de tiempo.

Paso 1. Calibrar la cámara o declarar el modelo virtual utilizado. Guardar resolución, intrínsecos, distorsión y transformación entre cámara y vehículo.

Paso 2. Definir características visuales estables y una política de asociación. El controlador debe recibir coordenadas y una medida de calidad, no únicamente una detección sin contexto.

Paso 3. Implementar la matriz de interacción y validarla con desplazamientos pequeños conocidos. Una diferencia de signo produce movimiento en dirección opuesta.

Paso 4. Ejecutar primero un IBVS clásico y utilizarlo como referencia. Después incorporar dinámica, horizonte y restricciones del campo de visión.

Paso 5. Definir márgenes internos de visibilidad y estados de degradación ante oclusión, baja confianza o retraso. La IA puede ayudar a generar detectores, pruebas sintéticas y adaptadores de cámara, pero

la seguridad no debe depender de una respuesta generativa en tiempo real.

Paso 6. Comparar ambos controladores con la misma secuencia de imágenes y reportar visibilidad, error en píxeles, comandos, latencia y pérdidas de objetivo.

11.12 Resumen del capítulo

El servo visual transforma errores de imagen en movimiento. La matriz de interacción conecta características y velocidad de cámara. Al unirla con una dinámica DMDc y un MPC, es posible anticipar la evolución visual y mantener los puntos dentro del campo de visión, una capacidad que el IBVS clásico no garantiza.

11.13 Preguntas de repaso

1. ¿Qué variable diferencia a IBVS de PBVS?
2. ¿Por qué la profundidad aparece en la matriz de interacción?
3. ¿Qué ventaja aporta incluir la dinámica DMDc?
4. ¿Por qué se recomienda un margen interior al borde de la imagen?
5. ¿Qué acción segura debe ejecutarse si se pierden las características?

12. Implementación y validación experimental

Objetivos del capítulo. Organizar una estrategia de validación progresiva, integrar software y hardware, verificar tiempo real, definir métricas y criterios de seguridad y documentar un experimento reproducible. Cada nivel debe tener una puerta de aceptación medible;

avanzar sin ella traslada incertidumbres de software al entorno de vuelo.

12.1 Validar por niveles

Un algoritmo no debe pasar directamente de una ecuación a un vuelo. Cada nivel de validación añade realismo y riesgo. Los niveles considerados incluyen MIL, Software-in-the-Loop (SIL), HIL y vuelo real. La Figura 30 muestra dicha escalera de validación y la Tabla 14 compara sus niveles.

Figura 30.

Escalera de validación desde el análisis matemático hasta el vuelo real

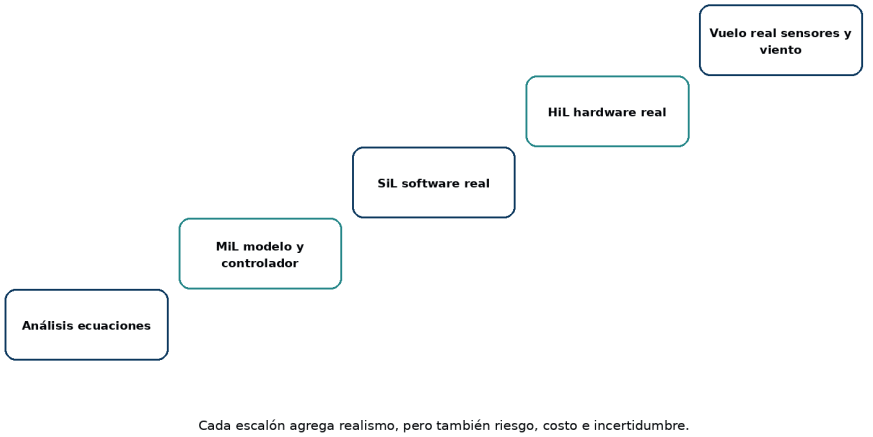


Tabla 14. *Niveles de validación*

Nivel	Planta	Controlador	Qué permite comprobar	Riesgo
Análisis	Ecuaciones	Ecuaciones	Unidades, equilibrio, estabilidad	Muy bajo

Nivel	Planta	Controlador	Qué permite comprobar	Riesgo
MIL	Modelo simulado	Modelo del algoritmo	Lógica, trayectoria, restricciones	Bajo
SIL	Simulador	Código final o cercano al final	Implementación y comunicaciones internas	Bajo
HIL	Planta simulada en tiempo real	Hardware y software reales	Tiempos, SDK, red, fallos de integración	Medio
Vuelo real	Vehículo físico	Sistema completo	Robustez y desempeño operacional	Alto

MIL se utilizó en el estudio de dinámica no modelada; Webots fue el entorno principal del servo visual; HIL permitió evaluar DMDc en línea; y los estudios comparativos de hexacóptero y de contorno añadieron vuelos reales.

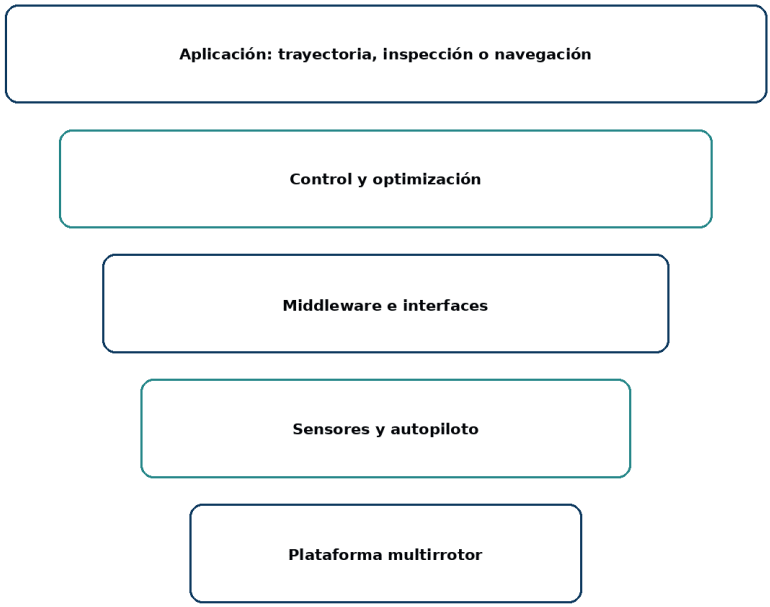
12.2 Arquitectura de software y hardware

Una implementación típica combina varios procesos:

- Nodo de sensores y estado
- Nodo de referencia o misión
- Nodo de identificación
- Nodo de control
- Solver de optimización
- Interfaz con el SDK
- Registro y visualización

Figura 31.

Pila de software y hardware para simulación, HIL y vuelo real



ROS facilita intercambio entre nodos. CasADi construye expresiones, derivadas y problemas de optimización. Webots o el simulador de DJI representan la planta. OpenCV procesa imágenes. El kit de desarrollo de software a bordo (Onboard Software Development Kit, OSDK) del fabricante entrega estado y recibe órdenes. La Figura 31 resume la pila de software y hardware.

La modularidad no debe convertirse en latencia innecesaria. Cada mensaje necesita una marca de tiempo; las tasas deben conocerse; y la cola de comunicación debe configurarse para evitar el uso de datos antiguos.

12.3 Presupuesto temporal

Para una tasa de 30 Hz se dispone de 33.3 ms por ciclo. Un presupuesto hipotético puede ser:

- Lectura y sincronización: 2 ms
- Estimación o procesamiento visual: 6 ms
- Optimización: 18 ms
- Publicación y comunicación: 2 ms
- Margen: 5.3 ms

El promedio no basta. Se recomienda registrar el tiempo de cada bloque y calcular máximo, percentiles 95 y 99, número de incumplimientos y duración de la peor racha. Si el solver excede el período, el sistema debe decidir entre reutilizar la última acción, aplicar una política de respaldo o detener la misión.

12.4 Elección del hardware

Los seis estudios ilustran distintas plataformas. La comparación de controladores utiliza una Jetson Nano; el hexacóptero emplea un computador con procesador i7; el NMPCC se ejecuta en una Jetson Orin NX; y el servo visual se valida en simulación. El hardware debe seleccionarse después de perfilar el algoritmo, no solo por la cantidad de núcleos.

Aspectos relevantes:

- Rendimiento por núcleo del solver
- Memoria y ancho de banda
- Consumo y disipación térmica
- Masa y efecto sobre autonomía
- Compatibilidad con SDK y sistema operativo

- Aceleración mediante unidad de procesamiento gráfico (GPU) para visión, si realmente se utiliza
- Comportamiento ante reducción térmica de frecuencia

12.5 Instrumentación y registro

Un registro mínimo debe incluir:

- Referencia y estado medido
- Entrada calculada y entrada enviada
- Tiempo de cálculo y estado del solver
- Estimaciones de perturbación
- Distancias o valores de barrera
- Coordenadas de características visuales
- Nivel de batería y eventos de seguridad
- Marcas de tiempo de todos los mensajes

Registrar solo la trayectoria 3D impide diagnosticar saturaciones, retardos o fallos de optimización. Los datos deben guardarse en una frecuencia suficiente y con una convención de unidades documentada.

12.6 Métricas

Las métricas deben responder a la misión. Entre las más útiles se encuentran:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{k=1}^{n} \| \mathbf{e}_k \|^2},$$

$$\mathrm{ISE} = \sum_{k=1}^{n} \| \mathbf{e}_k \|^2 \, \Delta t,$$

además de error máximo, tiempo de establecimiento, esfuerzo de control, distancia mínima, porcentaje de características visibles, tiempo del solver y tasa de problemas inviables.

Una sola cifra puede ocultar un evento peligroso. Un RMSE bajo es compatible con un pico grande y breve. Por ello, deben reportarse medidas medias y extremas.

12.7 Preparación del vuelo

Antes de despegar:

1. Verificar sentido de ejes y signo de cada orden.
2. Probar saturaciones con hélices retiradas cuando corresponda.
3. Confirmar frecuencia y marcas de tiempo.
4. Validar retorno a modo manual o aterrizaje.
5. Definir un volumen de vuelo y una distancia de seguridad.
6. Revisar viento, batería y fijación del computador.
7. Iniciar con referencias pequeñas y aumentar gradualmente.
8. Mantener un observador con capacidad de abortar.

Advertencia. El software de seguridad debe ser más sencillo que el algoritmo experimental. Una política de respaldo compleja puede fallar por las mismas razones que el controlador principal.

12.8 Reproducibilidad

Un experimento reproducible describe:

- Versión del modelo y del código
- Parámetros del vehículo

- Frecuencia y horizonte
- Matrices de peso y límites
- Condiciones iniciales y referencia
- Hardware, solver y tolerancias
- Proceso de calibración
- Criterios de exclusión de pruebas
- Datos y scripts para generar métricas

El propósito no es que dos laboratorios obtengan números idénticos, sino que puedan reconstruir las decisiones y explicar diferencias.

12.9 Guía de implementación general con apoyo de IA

La validación se automatiza mediante una única configuración que pueda ejecutarse en MIL, Software-in-the-Loop (SIL), HIL y vuelo, cambiando solo el adaptador de la planta.

Paso 1. Definir interfaces de estado, comando, referencia y diagnóstico. Cada mensaje debe incluir unidad, marco y marca de tiempo.

Paso 2. Crear un lanzador de experimentos que cargue configuración, asigne un identificador único y guarde versiones de software y parámetros.

Paso 3. Incorporar comprobaciones previas: límites, frecuencia, sensor congelado, batería, estado del solver y comunicación.

Paso 4. Automatizar métricas y gráficas desde los datos crudos. La IA puede ayudar a redactar scripts y detectar anomalías, pero el resultado debe obtenerse de nuevo sin intervención manual.

Paso 5. Establecer puertas de aceptación entre niveles y una política de aborto independiente. Solo se avanza cuando se cumplen estabilidad, tiempo real y seguridad.

Paso 6. Pedir a otra persona que ejecute el procedimiento utilizando únicamente el repositorio y la documentación; las dudas que aparezcan se convierten en mejoras del paquete de réplica.

12.10 Resumen del capítulo

La validación progresiva reduce riesgo y descubre fallos de integración. MIL, SIL, HIL y vuelo real responden preguntas distintas. Una arquitectura modular, un presupuesto temporal, un registro completo y un plan de seguridad son tan importantes como la ley de control.

12.11 Preguntas de repaso

1. ¿Qué diferencia práctica existe entre SIL y HIL?
2. ¿Por qué debe registrarse la entrada enviada además de la calculada?
3. ¿Qué información aporta el percentil 99 del tiempo del solver?
4. ¿Qué métricas complementan al RMSE?
5. ¿Qué características debe tener una política de respaldo?

13. Proceso integrado y decisiones de diseño

Objetivos del capítulo. Integrar modelación, identificación, control, percepción y validación en un flujo único; seleccionar métodos según la misión; reconocer errores frecuentes y utilizar el contenido como una base coherente para nuevos desarrollos. El flujo propuesto sirve como criterio de trazabilidad: cada decisión debe justificarse por una hipótesis, una medición o una restricción de misión.

13.1 Comenzar por requisitos

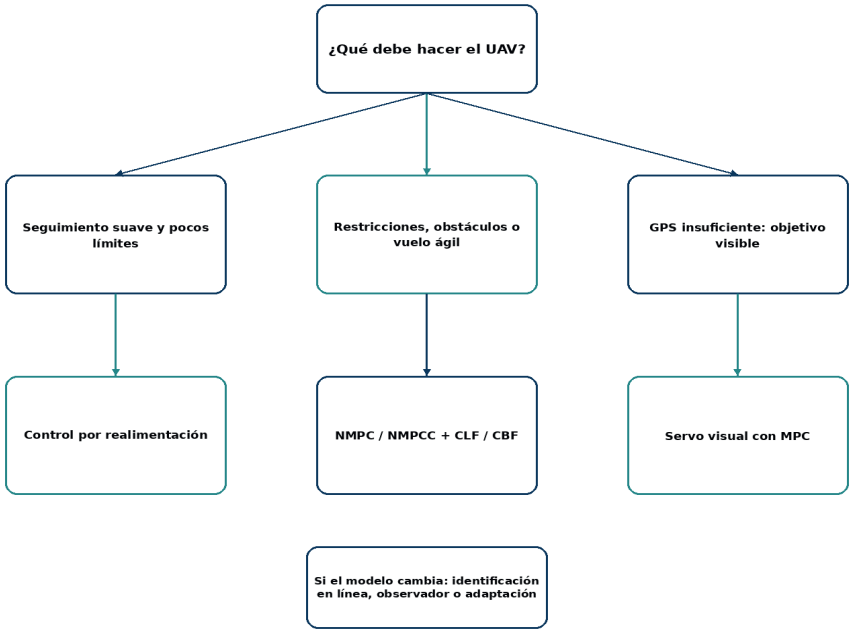
El diseño debe empezar con preguntas concretas:

- ¿Se controla posición, camino, imagen o interacción?
- ¿Qué sensores están disponibles?
- ¿Qué órdenes acepta la plataforma?
- ¿Qué límites son duros?
- ¿Qué perturbaciones se esperan?
- ¿Qué frecuencia y hardware existen?
- ¿Cuál es el criterio de seguridad?

Sin estas respuestas, elegir DMDc, NMPC o CBF es prematuro. La Figura 32 presenta el flujo práctico de diseño.

Figura 32.

Flujo práctico desde los requisitos hasta la validación y la mejora iterativa



13.2 Paso 1: definir variables y marcos

Se documentan marco inercial, marco del cuerpo, cámara y punto de interés. Se eligen estados medibles y entradas reales. En una DJI Matrice comercial, el vector reducido $[x, y, z, \psi, v_l, v_m, v_n, \omega]$ puede ser más útil que un modelo de motores inaccesible.

13.3 Paso 2: seleccionar el nivel de modelo

La decisión depende de la tarea:

- Modelo cinemático para seguimiento lento y diseño inicial
- Modelo físico reducido para compensación y análisis
- DMDc offline para mejorar predicción en el régimen observado
- DMDc online para cambios graduales
- Perturbación equivalente y observador para discrepancias rápidas
- Modelo de cámara aumentado para servo visual

El modelo debe validarse con señales diferentes de las utilizadas para identificarlo. Si solo reproduce el conjunto de entrenamiento, no se conoce su capacidad de predicción. La Tabla 15 orienta la selección del controlador según el escenario.

13.4 Paso 3: elegir el controlador

Tabla 15. *Selección de controlador por escenario*

Escenario	Opción inicial	Extensión recomendada	Razón
Vuelo suave sin obstáculos	Cinemática inversa	Compensación dinámica	Simplicidad y alta frecuencia

Escenario	Opción inicial	Extensión recomendada	Razón
Trayectoria rápida	Compensación dinámica	NMPC	Anticipación y mejor seguimiento
Límites múltiples	NMPC	Restricciones blandas y CLF	Manejo explícito de viabilidad
Cambios de carga	DMDc en línea	Adaptación u observador	Actualización de la dinámica
Obstáculos conocidos	NMPC	CBF/NMPCC	Seguridad dentro del OCP
Camino con libertad temporal	NMPCC	CLF + CBF	Separación de geometría y progreso
Aproximación visual	IBVS	MPC con FOV y dinámica	Mantenimiento de características visibles

13.5 Paso 4: separar desempeño y seguridad

La función de costo expresa preferencias; las restricciones, obligaciones. Un error frecuente es intentar garantizar seguridad solo con un peso muy alto. Un peso puede ser sacrificado si otro término domina; una CBF o una restricción de distancia representa explícitamente el conjunto seguro.

La seguridad también necesita información. Si la posición del obstáculo es incierta, el margen debe incorporar esa incertidumbre. Si una característica visual puede desaparecer, el controlador necesita una estrategia de recuperación. Si el estado llega con retardo, la predicción debe compensarlo o reducir la velocidad.

13.6 Paso 5: construir una validación incremental

Cada función nueva se activa después de comprobar la anterior:

1. Modelo y unidades
2. Control sin restricciones
3. Límites de entrada
4. Perturbaciones
5. Obstáculo estático
6. Obstáculo móvil
7. Ruido y retardo
8. HIL
9. Vuelo lento
10. Misión completa

La secuencia permite atribuir un fallo al cambio que lo introdujo.

13.7 Errores frecuentes

Confundir marcos

Enviar una velocidad inercial como si estuviera expresada en el cuerpo produce movimientos rotados. La corrección es documentar cada vector con superíndice o subíndice de marco.

Identificar con datos poco informativos

Un vuelo casi estacionario no permite separar parámetros. Deben diseñarse señales seguras que exciten cada eje.

Ajustar el controlador para compensar un modelo incorrecto

Aumentar ganancias puede ocultar temporalmente una dinámica mal identificada y generar saturación. Primero se valida la predicción.

Ignorar el tiempo total

Un solver de 20 ms puede fallar en un ciclo de 30 Hz si visión y comunicación consumen otros 20 ms. El presupuesto es de extremo a extremo.

Utilizar restricciones incompatibles

Pedir mantener el camino, conservar una distancia grande y no superar una velocidad pequeña puede dejar el problema sin solución. Las holguras y prioridades deben diseñarse conscientemente.

Validar solo casos exitosos

Un algoritmo seguro se prueba con pérdida de sensor, inicialización errónea, viento, batería baja, dato retrasado y problema de optimización fallido.

13.8 Arquitectura recomendada para una nueva investigación

Una plataforma de investigación general puede organizarse así:

1. Estimador de estado con marcas de tiempo
2. Modelo reducido validado offline
3. Identificación online opcional con vigilancia de calidad
4. Controlador base de bajo costo como respaldo
5. NMPC o NMPCC como controlador principal
6. Módulo de seguridad independiente
7. Percepción visual o de obstáculos con medida de incertidumbre
8. Registro reproducible y monitor de tiempo real

La arquitectura permite sustituir componentes sin rehacer el sistema completo. Por ejemplo, puede cambiarse DMDc por SINDy o un observador por otro estimador, manteniendo interfaces y pruebas.

13.9 Agenda de desarrollo

Las líneas futuras más directas son:

- Modelos híbridos que combinen física y aprendizaje
- Adaptación con garantías y detección de datos no informativos
- CBF robustas ante incertidumbre de obstáculos
- Percepción multimodal y seguimiento visual de objetivos móviles
- Solvers embebidos con certificación temporal
- Transferencia sim-to-real con cuantificación de discrepancia
- Control compartido y supervisión humana
- Coordinación de múltiples multirrotores

El aprendizaje automático puede mejorar percepción o modelación, pero no elimina la necesidad de restricciones, análisis de unidades y validación segura.

13.10 Resumen del capítulo

El diseño integrado comienza con la misión y las interfaces reales. Después selecciona un modelo, lo valida, elige el controlador más sencillo que cumpla los requisitos, separa desempeño de seguridad y avanza por niveles de validación. Se forma una secuencia coherente desde modelación e identificación hasta percepción y navegación segura.

13.11 Preguntas de repaso

1. ¿Por qué las entradas accesibles condicionan el modelo?
2. ¿Qué diferencia existe entre una preferencia de costo y una obligación de seguridad?
3. ¿Qué controlador utilizaría como respaldo de un NMPC?
4. ¿Por qué conviene activar funciones complejas de manera incremental?

14. Guía de reproducción y transferencia experimental

Objetivos del capítulo. Convertir las formulaciones de los capítulos anteriores en experimentos que otro equipo pueda repetir; identificar la información mínima que debe conservarse; definir protocolos de identificación y control comparables; y establecer criterios de aceptación antes de trasladar un algoritmo desde la simulación hasta el vuelo real. La transferencia responsable exige repetir primero las condiciones verificables y documentar explícitamente aquello que cambia entre plataformas, sensores o entornos.

La reproducibilidad no consiste únicamente en publicar un programa que pueda ejecutarse. Un resultado es reproducible cuando otra persona puede reconstruir la cadena completa de decisiones que lo produjo: qué plataforma se utilizó, cómo estaban definidos los ejes, qué señales se registraron, cómo se sincronizaron, qué modelo se identificó, qué parámetros se ajustaron, qué restricciones estuvieron activas, cuánto tardó cada ciclo y bajo qué condiciones se consideró exitoso el experimento.

En robótica aérea, esta exigencia es especialmente importante. Dos vehículos con una apariencia semejante pueden responder de manera distinta debido a la masa, la ubicación del centro de gravedad, las hélices, la batería, el firmware, los filtros internos o los límites impuestos por el fabricante. Incluso el mismo vehículo puede cambiar cuando se instala una cámara, disminuye la carga de la batería o aparece viento. Por ello, la réplica debe separar con claridad aquello que proviene de un artículo científico, aquello que se adapta a una nueva plataforma y aquello que constituye una decisión del investigador.

Este capítulo propone una guía práctica para reproducir los seis tipos de resultados empleados en la obra: identificación DMDc fuera de línea, identificación DMDc en línea, controladores de realimentación, NMPC con restricciones, estimación de dinámica no modelada, seguimiento seguro mediante NMPCC y servo visual predictivo. Los valores numéricos reportados se utilizan como referencias de orden de magnitud, no como parámetros universales. Cada implementación debe volver a medir, identificar y validar.

14.1 Reproducción de un resultado de control

Conviene distinguir tres niveles. El primero es la reproducción computacional. En este nivel, otra persona ejecuta el mismo código con los mismos datos y obtiene resultados equivalentes. Es el nivel más básico y exige conservar el programa, las dependencias, los archivos de configuración, las semillas aleatorias y los datos sin modificar. Una figura que solo puede recrearse copiando valores manualmente no satisface este nivel.

El segundo es la reproducción experimental. Otro equipo implementa el método en una plataforma equivalente y observa el mismo comportamiento cualitativo: convergencia, mejora respecto al

controlador base, respeto de restricciones y tiempos compatibles con la frecuencia de control. Los valores exactos pueden cambiar, pero las conclusiones principales deben mantenerse. Este nivel requiere describir el hardware, la calibración, las condiciones iniciales y los procedimientos de seguridad.

El tercero es la transferencia. El algoritmo se aplica en una plataforma o misión diferente. Aquí no se espera repetir cada número, sino demostrar que el procedimiento de diseño conduce nuevamente a un sistema funcional. La transferencia exige volver a identificar el modelo, ajustar límites y verificar el presupuesto temporal. Es el nivel más útil desde el punto de vista de ingeniería, pero también el que requiere mayor documentación.

Para evitar ambigüedades, cada resultado del libro debería asociarse con una afirmación verificable. Por ejemplo, la afirmación de que DMDc aproxima mejor la dinámica que el modelo paramétrico en los ejes evaluados se comprueba con datos independientes y una métrica definida. Que el NMPC es más preciso que los controladores base se constata usando la misma trayectoria, las mismas condiciones iniciales y el mismo criterio de error. Que la CBF mantiene seguridad se verifica registrando la distancia mínima y verificando que el conjunto seguro no se abandone. Que el servo visual conserva el objetivo en la imagen se comprueba midiendo el margen respecto a los límites del campo de visión.

Idea clave. La unidad mínima de reproducción no es una gráfica, sino una afirmación acompañada de datos, configuración, código, métrica y criterio de aceptación.

Antes de iniciar una réplica, escriba las afirmaciones que desea verificar. Después, para cada una, identifique la entrada, la salida, la métrica y la condición de éxito. Este ejercicio evita recopilar grandes

cantidades de datos que luego no responden a ninguna pregunta concreta.

14.2 Manifiesto del experimento

Todo ensayo debe comenzar con un manifiesto legible por humanos y por programas. Puede almacenarse en YAML (YAML Ain't Markup Language), JSON (JavaScript Object Notation) o TOML (Tom's Obvious Minimal Language). El manifiesto describe la identidad del experimento y evita que la información se disperse entre nombres de carpetas, mensajes de correo y notas personales.

Como mínimo, el manifiesto debe incluir: identificador del ensayo; fecha y zona horaria; plataforma; masa total; carga útil; batería; firmware; computadora de a bordo; sistema operativo; versiones de MATLAB, Python, ROS, CasADi, acados o Webots; frecuencia nominal de cada proceso; sensores utilizados; marcos de referencia; unidades; trayectoria; controlador; archivo de parámetros; conjunto de restricciones; criterio de terminación; condición meteorológica; responsable; versión del código; y ruta de los datos crudos.

También debe distinguir los parámetros declarados de los parámetros medidos. La frecuencia nominal puede ser 30 Hz, pero la frecuencia efectiva puede presentar variaciones. La masa indicada por el fabricante puede no incluir accesorios. La distancia a un obstáculo puede proceder de un plano conocido o de un sensor con incertidumbre. Mantener ambas columnas facilita la interpretación de diferencias entre laboratorios.

Un ejemplo compacto, que utiliza las convenciones este-norte-arriba (ENU) y frente-izquierda-arriba (FLU), es el siguiente:

```
experiment_id: nmpc_m100_lissajous_001
platform: DJI_Matrice_100
payload_kg: 0.62
controller: nmpc
model: dmdc_offline_v3
sample_time_nominal_s: 0.0333
trajectory: lissajous_low_speed
frames:
  inertial: ENU
  body: FLU
units:
  position: m
  velocity: m_s
  yaw: rad
software:
  os: registrar_version_exacta
  casadi: registrar_version_exacta
  solver: registrar_nombre_y_version
git_commit: registrar_hash
raw_data: data/raw/nmpc_m100_lissajous_001
status: completed
```

El manifiesto no debe modificarse después del ensayo sin dejar una traza. Si se descubre un error, se crea una revisión y se documenta el cambio. Los datos crudos se consideran inmutables. El filtrado, la interpolación y el recorte generan nuevos archivos derivados; nunca sustituyen al registro original.

Para cada sesión de vuelo conviene producir además una hoja breve de incidencias. Debe registrar cambios de batería, golpes, reinicios, pérdida de GPS, saturación, intervención del piloto,

modificación de la trayectoria o cambio de configuración. Una observación escrita durante el experimento puede explicar un pico que no es visible en el registro digital.

Advertencia. Dos archivos con el mismo nombre no son necesariamente el mismo experimento. Utilice identificadores únicos, control de versiones y sumas de verificación para datos y configuraciones.

14.3 Organización del proyecto y del entorno de software

Una estructura de carpetas estable reduce errores y facilita automatizar el procesamiento. El código, los datos y los resultados deben estar separados. Los archivos de configuración deben ser pequeños y legibles. Las figuras del informe deben generarse directamente desde los datos procesados mediante un script, de modo que no exista una edición manual imposible de rastrear.

```
proyecto_uav/
  README.md
  CITATION.cff
  environment/
    requirements.txt
    environment.yml
    container.def
  config/
    platform/
    trajectories/
    controllers/
    experiments/
  data/
    raw/
```

```
  interim/
  processed/
 models/
  identified/
  validation/
 src/
  common/
  identification/
  controllers/
  perception/
  safety/
 tests/
  unit/
  integration/
  regression/
 scripts/
  run_identification.py
  run_simulation.py
  run_hil.py
  generate_figures.py
 results/
  metrics/
  figures/
  logs/
```

El archivo README debe explicar cómo reconstruir el entorno desde cero. Es preferible indicar comandos completos a escribir únicamente "instale las dependencias". Cuando sea posible, se utiliza un entorno aislado o un contenedor. Sin embargo, el contenedor no reemplaza la documentación del hardware: un solver puede comportarse de manera distinta en dos procesadores, aunque el software sea idéntico.

Las versiones deben fijarse. Una actualización de CasADi, acados, Numerical Python (NumPy), Eigen o el compilador puede modificar convergencia, tolerancias o tiempo de ejecución. En sistemas con ROS también deben registrarse la distribución, los tipos de mensajes y la calidad de servicio. Si se utiliza MATLAB, se conservan la versión, los toolboxes y las opciones del solver. En Webots se guarda el mundo, los archivos de prototipo de Webots (PROTO), el paso básico de simulación y los controladores.

El proyecto debe incluir pruebas automáticas sencillas. Una prueba unitaria verifica, por ejemplo, que una rotación de 90 grados transforma correctamente una velocidad. Una prueba de integración comprueba que el controlador y el simulador intercambian mensajes. Una prueba de regresión ejecuta un escenario corto y compara métricas con un intervalo esperado. Estas pruebas no sustituyen el vuelo, pero detectan cambios accidentales antes de usar la plataforma.

Para preservar la trazabilidad, cada ejecución debe copiar al directorio de resultados el archivo de configuración exacto y el identificador de la versión de código. No es suficiente conservar el archivo actual del repositorio, porque puede haber cambiado después. Una práctica útil es generar automáticamente un archivo run_info.json con versiones, hora de inicio, nombre del equipo, número de núcleos, memoria disponible y variables relevantes del sistema.

14.4 Tiempo, sincronización y calidad del registro

La mayoría de los problemas de identificación que parecen matemáticos son, en realidad, problemas de tiempo. Si la referencia de velocidad y la respuesta del vehículo están desplazadas en algunas muestras, el modelo aprende una relación incorrecta. Si cada sensor utiliza un reloj diferente, la fusión puede introducir retardos variables. Por

eso, toda señal debe llevar una marca de tiempo generada lo más cerca posible del punto de medición.

Se recomienda utilizar un reloj monótono para calcular intervalos y un reloj de calendario para identificar la sesión. El reloj monótono no cambia si el sistema corrige la hora. Cuando intervienen varias computadoras, deben sincronizarse mediante un mecanismo apropiado y medir el error residual. En HIL, además, se debe definir si el simulador funciona en tiempo real, más rápido o más lento que el tiempo físico.

El registro debe incluir tanto la referencia solicitada como la aceptada por la interfaz. Algunos SDK saturan o filtran internamente las órdenes. Si solo se almacena la orden original, el modelo puede atribuir al vehículo una dinámica que en realidad pertenece al autopiloto. También deben registrarse el estado estimado, las mediciones crudas disponibles, el estado del controlador, las restricciones activas, el tiempo del solver y cualquier comando de seguridad.

Una fila mínima para identificación de alto nivel podría contener: timestamp_ns, experiment_id, mode, x_m, y_m, z_m, yaw_rad, u_m_s, v_m_s, w_m_s, r_rad_s, u_ref_m_s, v_ref_m_s, w_ref_m_s, r_ref_rad_s, solver_status, solver_time_ms, battery_v, gps_quality, wind_estimate_m_s, safety_state

No se deben mezclar grados con radianes ni segundos con milisegundos. Las unidades se incorporan al nombre del campo o al diccionario de datos. Los signos de los ejes se verifican con movimientos simples. Si se utilizan convenciones ENU, norte-este-abajo (NED), FLU o frente-derecha-abajo (FRD), deben declararse explícitamente.

Durante el procesamiento, primero se representa el intervalo real entre muestras. Después se decide si es necesario remuestrear. La

interpolación puede facilitar el cálculo, pero también ocultar paquetes perdidos. Por ello, se conserva una máscara que indica qué valores fueron medidos y cuáles interpolados. Los segmentos con pausas largas, reinicios o cambios de modo se separan antes de identificar.

Las derivadas amplifican ruido. Para obtener aceleraciones se pueden utilizar diferencias centrales, filtros de Savitzky-Golay, observadores o modelos de estado. El método y sus parámetros deben formar parte de la configuración. No debe utilizarse el mismo filtro para todos los ejes por costumbre; el ruido y la dinámica pueden diferir.

Ejemplo. Si el controlador trabaja a 30 Hz, el período nominal es aproximadamente de 33.3 ms. La réplica debe informar también la mediana, el percentil 95 y el máximo del período observado. Una media cercana a 33.3 ms puede ocultar ciclos ocasionales de 80 ms que afectan la estabilidad.

14.5 Reproducción del modelo cinemático

La cinemática reducida es el primer componente que debe verificarse, porque todos los controladores dependen de ella. La prueba comienza con el vehículo inmóvil y una orientación conocida. Se envían referencias pequeñas, una por una, para avance, movimiento lateral, vertical y guiñada. El objetivo no es ajustar ganancias, sino comprobar sentidos, escalas y marcos.

Para cada eje se registra la referencia, la velocidad estimada y la derivada de la posición. Si el vehículo tiene yaw igual a cero, una referencia positiva de avance debe producir movimiento en el eje inercial esperado. Después se repite con yaw cercano a 90 grados. La comparación permite verificar la matriz de rotación. Una inversión de signo o un intercambio de ejes debe resolverse en una capa de interfaz, no ocultarse dentro del controlador.

Si se controla un punto desplazado, como una cámara, debe medirse el vector desde el centro de masa hasta el punto de interés. La medición se expresa en el marco del cuerpo y se conserva con su incertidumbre. Durante una rotación pura de yaw, el punto desplazado debe recorrer un arco. Esta prueba permite confirmar los términos adicionales del jacobiano.

El procedimiento recomendado es:

1. Definir los marcos mediante un diagrama y una fotografía de la plataforma.
2. Medir la orientación inicial y verificar la convención de yaw.
3. Enviar escalones pequeños y seguros en cada canal.
4. Calcular la predicción cinemática usando las señales realmente aceptadas por el SDK.
5. Comparar la derivada medida y la predicha.
6. Repetir para varias orientaciones.
7. Documentar cualquier transformación necesaria entre la interfaz y el modelo.

La aceptación no debe basarse únicamente en que la trayectoria “se ve correcta”. Se calculan error medio, error máximo y correlación por canal. También se inspeccionan retardos. Si la dirección es correcta, pero la amplitud difiere, puede existir un escalado o una limitación interna. Si la amplitud es correcta pero la señal está desplazada, debe estimarse la latencia.

Para comprobar la implementación discreta, se ejecuta el modelo con el mismo período de las mediciones y se compara con una integración usando un paso menor. Si las diferencias son relevantes, se reduce el paso interno o se usa Runge-Kutta. En modelos con

cuaterniones se verifica que la norma permanezca cerca de uno y se normaliza de forma controlada.

La cinemática se considera lista para los capítulos de control cuando: los cuatro canales tienen signos correctos; las unidades coinciden; el retardo está caracterizado; el punto de interés está definido; y una simulación abierta reproduce la evolución de la pose durante maniobras suaves.

14.6 Reproducción de la identificación fuera de línea

La identificación debe comenzar con una pregunta: ¿qué rango de operación debe representar el modelo? Un conjunto de datos de vuelo estacionario no sirve para predecir una trayectoria rápida. Del mismo modo, una señal agresiva puede ser innecesaria y peligrosa si la misión final es lenta. La excitación se diseña para cubrir las velocidades, aceleraciones y combinaciones de ejes esperadas.

Las señales pueden ser escalones suaves, secuencias pseudoaleatorias limitadas, *chirps* o trayectorias compuestas. Deben respetar las restricciones de la plataforma. Para reducir el acoplamiento durante la primera identificación, se excita un canal por vez; después se realizan maniobras combinadas para verificar que el modelo capture interacciones. Cada ensayo debe incluir un período inicial de estabilización y un período final que permita observar la respuesta.

Los datos se dividen por vuelos completos en entrenamiento, validación y prueba. Dividir aleatoriamente muestras consecutivas produce fuga de información, porque puntos casi idénticos aparecen en ambos conjuntos. El conjunto de prueba se reserva hasta el final y utiliza señales diferentes. Si se comparan Euler-Lagrange y DMDc, ambos modelos deben entrenarse y evaluarse con los mismos segmentos.

Antes de identificar se realizan las siguientes comprobaciones:

- Continuidad de las marcas de tiempo
- Ausencia de cambios de modo no documentados
- Límites y saturaciones
- Coherencia de unidades
- Correspondencia entre referencia y respuesta
- Suficiente variación en cada canal
- Ausencia de segmentos con intervención manual no declarada

En la formulación paramétrica, la matriz regresora debe construirse con la misma convención de marcos empleada en el controlador. Los parámetros se estiman con una función de costo definida y, cuando sea posible, con límites físicos. Un valor matemáticamente óptimo pero físicamente imposible suele indicar falta de excitación, error de signo o colinealidad.

En DMDc se forman las matrices de instantáneas. Para el modelo discreto se utiliza una relación del tipo X_plus = A X + B U. Los estados y entradas pueden normalizarse para evitar que una variable con gran magnitud domine la descomposición. La media y la escala empleadas se guardan junto con el modelo; de lo contrario, las matrices no pueden utilizarse correctamente en otra ejecución.

La selección del rango de la SVD no debe depender únicamente de un porcentaje de energía. Se evalúan varios rangos y se comparan error de validación, estabilidad de la predicción y condición numérica. Un rango muy bajo elimina dinámica relevante; un rango muy alto ajusta ruido. Cuando el modelo se utilizará dentro de un MPC, también se comprueba que no produzca predicciones divergentes durante el horizonte.

Guía de implementación para la identificación DMDc fuera de línea

1. Cargar el manifiesto del experimento y conservar una copia inmutable de los datos crudos.
2. Verificar unidades, marcos de referencia, orden de variables y marcas de tiempo.
3. Separar vuelos completos para entrenamiento, validación y prueba, evitando mezclar muestras consecutivas.
4. Aplicar únicamente el preprocesamiento declarado y guardar sus parámetros.
5. Construir las matrices de estados, estados futuros y entradas con dimensiones verificadas.
6. Calcular la descomposición en valores singulares y revisar su espectro y condición numérica.
7. Evaluar varios rangos candidatos en lugar de elegir uno por una regla automática única.
8. Estimar las matrices del modelo para cada rango candidato.
9. Comparar predicción a un paso, predicción libre a varios pasos y residuos.
10. Seleccionar el modelo que equilibre precisión, estabilidad y complejidad.
11. Guardar matrices, escalado, frecuencia, variables, versión del software y métricas.
12. Ejecutar el conjunto de prueba una sola vez y documentar tanto los resultados favorables como los fallos.

Las métricas mínimas son RMSE por estado, ISE, error máximo y error de predicción a varios pasos. También se representa el residuo en el tiempo. Un único número agregado puede esconder que el

modelo funciona en vertical pero falla en guiñada. La comparación con el modelo paramétrico debe indicar claramente si se predicen velocidades, aceleraciones o estados futuros.

El modelo publicado debe acompañarse de un archivo que contenga: nombres y orden de las variables, unidades, frecuencia, normalización, matrices, rango elegido, código para cargarlo, datos de validación o un subconjunto, y un script que genere las métricas. Sin esa información, las matrices aisladas son difíciles de reutilizar.

14.7 Reproducción de DMDc en línea y control adaptativo

La identificación en línea agrega una dificultad: el modelo cambia mientras controla el vehículo. Por ello, no basta con actualizar matrices en cada muestra. Se necesita una política que determine cuándo los datos son suficientemente informativos y cuándo el nuevo modelo es seguro.

El estudio de DMDc en línea utilizó como referencia una ejecución a 30 Hz y una ventana corta de seis muestras en un marco HIL. Estos valores muestran que la actualización puede ser rápida, pero no deben copiarse sin analizar la nueva plataforma. Una ventana corta responde pronto a cambios, aunque es sensible al ruido y puede tener rango insuficiente. Una ventana larga es más estable, pero adapta lentamente.

La réplica debe registrar, en cada actualización: tamaño de la ventana, rango, valores singulares, número de condición, residuo, matrices anteriores y nuevas, y razón para aceptar o rechazar el cambio. Una regla de aceptación puede exigir rango completo, condición menor que un umbral, residuo acotado y variación limitada respecto al modelo anterior.

Cuando la excitación es pobre, se conserva el último modelo válido. No se debe forzar una actualización durante vuelo estacionario si las columnas de datos son casi dependientes. También conviene proyectar o saturar parámetros para evitar saltos no plausibles. Un modelo actualizado puede probarse durante uno o varios pasos en modo sombra: genera predicciones, pero todavía no gobierna el controlador. Solo se activa si mejora la métrica reciente.

Guía de implementación para actualizar un modelo DMDc en línea

1. Insertar la nueva transición de estado y entrada en una ventana circular con marcas de tiempo.
2. Calcular rango, número de condición, valores singulares y residuo antes de modificar el modelo activo.
3. Estimar matrices candidatas únicamente cuando la ventana contenga información suficiente.
4. Limitar variaciones físicamente no plausibles y comprobar estabilidad numérica.
5. Comparar la predicción del modelo candidato con la del modelo activo durante un período de prueba o modo sombra.
6. Activar el modelo candidato solo cuando mejora una métrica definida y conserva las condiciones de seguridad.
7. Mantener el último modelo válido cuando la excitación sea insuficiente o la actualización no supere la puerta de aceptación.
8. Registrar la decisión, los diagnósticos y las matrices anterior y nueva para permitir auditoría.

Para evaluar adaptación se diseña una perturbación repetible. Puede cambiarse una carga conocida, variar un parámetro del simulador o aplicar una fuerza virtual. El ensayo incluye tres fases: modelo

nominal, cambio y recuperación. Se comparan un controlador con modelo fijo y uno con actualización. La mejora debe medirse en seguimiento y no solo en error de predicción.

La latencia de actualización forma parte del experimento. El cálculo DMDc y el controlador deben terminar dentro del período. Se informan mediana, percentil 95 y máximo. Si la actualización es más lenta que la ley de control, puede ejecutarse a una frecuencia inferior, manteniendo el controlador a la tasa principal.

Una práctica recomendable consiste en separar tres relojes: adquisición, actualización del modelo y control. Cada proceso tiene su propia marca de tiempo. Así se evita atribuir al algoritmo un retraso producido por la comunicación.

14.8 Reproducción de los controladores de referencia

Los controladores base son esenciales para interpretar un resultado avanzado. Sin una implementación cuidada del método de referencia, una mejora del NMPC puede deberse a una comparación injusta. En los estudios considerados, las referencias principales son el control cinemático inverso y la compensación dinámica inversa.

La reproducción comienza con la misma trayectoria, estado inicial, límites y frecuencia para todos los controladores. Se utiliza el mismo estimador de estado y el mismo preprocesamiento. Si un controlador recibe una señal filtrada y otro una cruda, la comparación mezclaría el efecto del filtro con el efecto de la ley de control.

El controlador cinemático se ajusta primero a baja velocidad. Las ganancias se incrementan hasta lograr una respuesta suficientemente rápida sin oscilación ni saturación persistente. Después se repite con trayectorias más exigentes. Se registran las acciones antes y después

de saturar. La saturación debe formar parte del algoritmo, no añadirse al final sin analizarla.

La compensación dinámica requiere un modelo identificado. Se prueba primero en lazo abierto para confirmar que la aceleración predicha tiene dirección y escala razonables. Después se activa con ganancias conservadoras. Si el modelo contiene una inversión numérica, se vigila la condición de la matriz. Una regularización puede ser necesaria cuando la identificación produce matrices cercanas a singularidad.

Para comparar controladores se recomienda usar al menos tres trayectorias:

1. Referencia suave y lenta, donde todos deberían funcionar.
2. Referencia con mayor velocidad o curvatura, donde aparecen diferencias.
3. Referencia con perturbación o cambio de carga, donde se evalúa robustez.

Cada trayectoria se repite varias veces y se alterna el orden de los controladores para reducir el efecto de batería o viento. En simulación se utilizan exactamente las mismas condiciones y semillas. En vuelo real se registra la condición ambiental y se evita comparar ensayos realizados con diferencias extremas.

Las métricas incluyen error por eje, error tridimensional, máximo, tiempo de establecimiento, suavidad de la entrada y tiempo de cálculo. Un controlador rápido no es necesariamente mejor si produce mayor error; un controlador preciso no es utilizable si incumple el período. La conclusión debe presentar ambos ejes del compromiso.

Advertencia. No ajuste cada controlador sobre una trayectoria diferente. La selección de ganancias puede usar un conjunto de

desarrollo, pero la comparación final debe emplear trayectorias y condiciones comunes que no hayan sido utilizadas para decidir los parámetros.

14.9 Reproducción de un NMPC en CasADi

La implementación de NMPC debe documentarse como un problema matemático y como un programa. Dos códigos pueden contener la misma función de costo y comportarse de forma distinta por la discretización, el escalado, las tolerancias, el inicializador o la forma de imponer restricciones.

El primer paso es declarar el vector de estado y el de control en el orden exacto. Después se define la dinámica continua y el integrador. Si se utiliza disparo múltiple, los estados en cada nodo son variables de decisión y se añaden restricciones de continuidad. El número de pasos y el período determinan el horizonte total. En los estudios de esta obra aparecen horizontes de aproximadamente un segundo y, para el hexacóptero, un período de 0.1 s con un horizonte mayor en algunas pruebas. La nueva implementación debe justificar su selección mediante precisión y tiempo de cálculo.

La función de costo se divide en términos: seguimiento, orientación, esfuerzo, variación de la entrada, progreso u holguras. Cada término se normaliza por un rango representativo antes de elegir pesos. Sin escalado, un error de posición expresado en metros y una velocidad angular pueden tener magnitudes muy diferentes. Los pesos no deberían compensar unidades mal condicionadas.

Las restricciones deben enumerarse una por una: límites de velocidad, límites de entrada, tasa de cambio, altura, campo de vuelo, distancia a obstáculos, norma del cuaternión, y restricciones terminales. Cada una se clasifica como dura o blanda. Las restricciones de seguridad críticas

suelen ser duras, pero una holgura cuidadosamente penalizada puede evitar que el solver falle cuando el estado inicial ya es ligeramente inviable.

El solver se configura con tolerancias, máximo de iteraciones y límite de tiempo. Se utiliza una solución inicial. En tiempo real, la solución anterior se desplaza un paso y sirve como *warm start*. El controlador debe definir qué hacer si el solver devuelve un estado distinto de éxito, supera el tiempo o genera una acción no finita.

Guía de implementación del ciclo de control NMPC

1. Leer el estado estimado y verificar la antigüedad, validez y unidades de sus datos.
2. Generar la referencia para todo el horizonte de predicción.
3. Actualizar obstáculos, límites y parámetros del modelo mediante una interfaz claramente definida.
4. Desplazar la solución del ciclo anterior para usarla como inicialización cálida.
5. Construir límites y parámetros sin alterar el orden documentado de estados y entradas.
6. Resolver el problema de optimización con un límite de tiempo menor que el período de control.
7. Comprobar estado del solver, residuos, restricciones y finitud de la solución.
8. Aplicar únicamente la primera acción después de verificar saturaciones y tasa de cambio.
9. Ejecutar la política de respaldo si la solución no es válida o llega tarde.
10. Registrar costo, iteraciones, tiempo total, restricciones activas y acción realmente enviada.

La política de respaldo puede mantener la última acción durante un intervalo muy corto, pasar al controlador cinemático, ordenar flotación o iniciar aterrizaje. La opción depende de la misión. Lo importante es que esté definida antes del vuelo y sea probada mediante fallos inducidos.

El ajuste se realiza por etapas. Primero, se ejecuta sin obstáculos y con referencias suaves. Segundo, se activan límites de entrada. Tercero, se añade tasa de cambio, perturbaciones y, finalmente, obstáculos. Si todas las funciones se activan al mismo tiempo, es difícil localizar la causa de una inviabilidad.

Para evaluar el solver, se registra el tiempo total, no solo la llamada de optimización. La construcción de parámetros, la transferencia de memoria y la conversión de mensajes también consumen tiempo. Se informa la distribución y no únicamente el promedio. El máximo observado debe ser menor que el período o existir una estrategia explícita para ciclos tardíos.

Una prueba de regresión útil conserva un estado inicial y una secuencia de referencias pequeñas. Después de cualquier cambio, el problema debe producir un costo y una primera acción dentro de tolerancias. Esta prueba detecta alteraciones en signos, índices o pesos.

14.10 Reproducción del NMPC adaptativo y del observador de momento

La dinámica no modelada puede representarse como una perturbación equivalente. El observador estima esa perturbación a partir de la diferencia entre el comportamiento esperado y el observado. La ley adaptativa modifica parámetros para reducir errores persistentes. Su combinación busca responder tanto a cambios lentos como a discrepancias rápidas.

La réplica debe implementar y comparar al menos cuatro variantes: NMPC nominal, NMPC con adaptación, NMPC con observador, y NMPC con adaptación más observador. Esta ablación permite atribuir la mejora. Si solo se muestra la estrategia completa, no se sabe qué componente produce el efecto.

La primera tarea es definir exactamente dónde entra la perturbación. Puede sumarse a la aceleración, a la entrada efectiva o a un estado aumentado. La elección debe coincidir con la ecuación implementada. Después se determina qué señales necesita el observador y si están medidas o estimadas. El uso de derivadas numéricas exige filtrado y añade fase.

Las ganancias se ajustan en simulación. Una ganancia alta produce respuesta rápida, pero amplifica ruido. Una ganancia baja filtra mejor, aunque estima tarde. Se utilizan perturbaciones conocidas: escalón de fuerza, cambio de masa o término omitido de actitud. Como la perturbación verdadera es conocida en simulación, puede medirse directamente el error de estimación.

La estimación debe acotarse. Si un fallo de sensor produce un residuo grande, el observador no debe generar una compensación ilimitada. También se limita la velocidad de cambio. Cuando la calidad del estado es insuficiente, se congela o reinicia el estimador de manera controlada.

El procedimiento de validación incluye:

1. Ejecutar el modelo nominal sin perturbación y verificar que la estimación permanezca cerca de cero.
2. Aplicar una perturbación conocida y medir tiempo de detección, error máximo y error estacionario.

3. Añadir ruido y comparar el compromiso entre velocidad y suavidad.
4. Repetir con perturbaciones no utilizadas durante el ajuste.
5. Integrar el estimador al NMPC y evaluar seguimiento.
6. Comprobar que el tiempo de cálculo permanece dentro del período.
7. Inducir pérdida temporal de datos y verificar el modo seguro.

El resultado debe presentar tanto el seguimiento como la estimación. Una trayectoria precisa no demuestra que el observador reconstruya correctamente la perturbación; puede existir compensación por otras vías. De forma inversa, una estimación aparentemente buena no es útil si introduce ruido en el control.

En una transferencia a vuelo real, la estimación se activa inicialmente en modo sombra. Se registra, pero no se aplica. Después se limita a una fracción de la compensación y se incrementa de forma gradual. Este procedimiento reduce el riesgo de que un error de signo provoque una acción brusca.

14.11 Reproducción del NMPCC con CLF y CBF

El control predictivo de contorno requiere una representación explícita del camino. Se define una curva parametrizada y una variable de progreso. Para cada estado se calcula un punto de referencia sobre la curva y se separan el error transversal o de contorno y el error longitudinal o de avance. La implementación debe indicar cómo se obtiene la derivada de la curva y cómo se mantiene el parámetro dentro de su dominio.

Si el camino se construye con puntos, se utiliza una interpolación suficientemente suave. Una curva con discontinuidades en la derivada genera referencias de orientación y velocidad no físicas. Los puntos y el método de interpolación se publican junto con el código.

El error de orientación se calcula con cuaterniones. Debe verificarse la convención de multiplicación y la elección del signo, porque **q** y -**q** representan la misma orientación. Antes de aplicar el mapa logarítmico se selecciona la representación que produce el giro más corto. Se añaden pruebas unitarias con rotaciones conocidas.

La CLF se implementa como una desigualdad que promueve la disminución de una función de energía o error. La CBF define el conjunto seguro. Para obstáculos circulares o esféricos, la función puede depender de la distancia al centro menos el radio combinado de obstáculo, vehículo y margen. Si la función tiene grado relativo mayor que uno, se construye una CBF de orden superior. La derivación debe conservarse en un documento simbólico o cuaderno verificable.

El margen de seguridad no se elige únicamente por el tamaño físico. Debe incluir error de localización, incertidumbre del obstáculo, latencia y distancia recorrida durante el tiempo de reacción. Para un obstáculo móvil, la predicción debe declarar el modelo utilizado. Si la velocidad se considera constante, se evalúan escenarios donde esa hipótesis se incumple.

Las CLF y CBF pueden entrar en conflicto con límites de entrada. Por ello, se analizan estados iniciales cercanos a la frontera y situaciones donde el camino pasa demasiado cerca del obstáculo. Las holguras se asignan según prioridad. Una práctica común es permitir una pequeña relajación de desempeño antes que relajar seguridad.

Escenarios mínimos para implementar y validar un NMPCC seguro

1. Camino recto sin obstáculos: comprobar error de contorno y progreso.
2. Camino curvo sin obstáculos: comprobar orientación, continuidad y velocidad de avance.
3. Obstáculo estático alejado del camino: confirmar que no modifica innecesariamente la misión.
4. Obstáculo estático sobre el camino: medir desvío, holgura y distancia mínima.
5. Dos obstáculos que formen un paso estrecho: evaluar viabilidad y prioridad de restricciones.
6. Obstáculo móvil con velocidad conocida: verificar el modelo de predicción.
7. Error de localización y retardo: incrementar incertidumbre y comprobar el margen de seguridad.
8. Estado inicial próximo a la frontera segura: verificar recuperación sin violar la barrera.

En tiempo real se registran el valor de cada CBF, la restricción más activa, la holgura, el progreso, los errores de contorno y avance, el estado del solver y la distancia real. La gráfica de trayectoria es insuficiente para demostrar seguridad. El valor mínimo de la función barrera y la distancia mínima deben acompañar el resultado.

La implementación con acados o CasADi debe especificar método de integración, discretización, algoritmo de programación cuadrática (QP), número de iteraciones y estrategia de inicialización. El hardware también se reporta, porque el tiempo depende de él. En la plataforma experimental del estudio se utilizó un DJI Matrice 100 y

una computadora Jetson Orin NX; una réplica con otro equipo debe volver a medir el presupuesto.

14.12 Reproducción del servo visual predictivo

El servo visual integra cámara, procesamiento de imagen, geometría y control. Un resultado reproducible comienza con la calibración. Se publican los parámetros intrínsecos, resolución, frecuencia, modelo de distorsión y transformación entre cámara y cuerpo. La calibración se verifica con un conjunto diferente de imágenes y se reporta el error de reproyección.

La característica visual debe estar definida. Puede ser el centro de un marcador, varias esquinas o una línea. El detector, el descriptor, los umbrales y la política de asociación se documentan. Si se utiliza un objetivo artificial, se publican sus dimensiones y una imagen imprimible. Si el objetivo es natural, se proporciona un conjunto de imágenes o video.

Las coordenadas pueden expresarse en píxeles o normalizadas. La matriz de interacción debe coincidir con esa elección. También se define el origen de la imagen y el sentido del eje vertical. Un error frecuente es utilizar una formulación con eje positivo hacia arriba en una imagen cuyo índice crece hacia abajo.

La profundidad puede medirse, estimarse o fijarse. Si se asume constante, se evalúa sensibilidad. Si procede de una cámara RGB-D, se caracteriza ruido, rango y alineación. Si se estima mediante geometría, se documenta el método y sus fallos. El controlador debe recibir también una medida de confianza.

El campo de visión se transforma en restricciones internas con un margen. El margen considera tamaño del objetivo, error del

detector y latencia. Mantener el centro dentro de la imagen no garantiza que todas las esquinas permanezcan visibles. Para objetivos extendidos, se restringen varios puntos o se incorpora su caja delimitadora.

La referencia visual se prueba, primero, con una cámara virtual y un objetivo estático. Después, se introducen cambios de profundidad, movimientos laterales, ruido, retraso y pérdida de detección. Se compara IBVS clásico con la formulación predictiva usando las mismas condiciones. Las métricas incluyen error visual, porcentaje de cuadros con objetivo visible, margen mínimo respecto al borde, tiempo de recuperación, velocidad del vehículo y tiempo de cálculo.

Guía de implementación del ciclo de servo visual predictivo

1. Capturar la imagen junto con su marca de tiempo y parámetros de cámara.
2. Corregir distorsión o aplicar explícitamente el modelo de cámara declarado.
3. Detectar las características y calcular una medida de calidad o confianza.
4. Asociar las características con la detección anterior y descartar correspondencias ambiguas.
5. Medir, estimar o aproximar la profundidad e indicar su incertidumbre.
6. Construir el error visual, la matriz de interacción y el estado aumentado.
7. Comprobar la edad de la imagen y la coherencia temporal con el estado del vehículo.

8. Resolver el MPC con márgenes internos del campo de visión y límites de velocidad.
9. Aplicar la primera orden solamente cuando la solución y la percepción sean válidas.
10. Ante pérdida del objetivo, detener el avance y activar una estrategia segura de búsqueda, flotación o aborto.

El entorno Webots debe conservar mundo, pose inicial, iluminación, cámara, textura y paso de simulación. Las pruebas deben repetirse con variaciones de iluminación y ruido para evitar que el resultado dependa de una escena ideal. La velocidad de simulación y la frecuencia de la cámara se registran por separado.

En vuelo real se agrega una barrera de seguridad independiente del control visual. Si el objetivo se pierde, la aeronave no debe continuar con la última orden de manera indefinida. Se define una máquina de estados: seguimiento, degradado, búsqueda, flotación y aborto.

14.13 Escalera de validación y puertas de aceptación

La transición a vuelo real debe producirse por niveles. Cada nivel tiene una pregunta y una puerta de aceptación. No se avanza porque "parece funcionar", sino porque se cumplen criterios registrados.

Nivel 1: Pruebas unitarias. Se verifican rotaciones, jacobianos, proyecciones, gradientes, dimensiones y saturaciones. Las derivadas analíticas se comparan con diferencias finitas. Una discrepancia aquí se corrige antes de ejecutar el solver.

Nivel 2: Model-in-the-Loop. El controlador y la planta son modelos matemáticos. Se evalúan trayectorias nominales, perturbaciones y

restricciones. La aceptación exige convergencia, ausencia de valores no finitos y tiempo compatible en el equipo de desarrollo.

Nivel 3: Software-in-the-Loop. El código que se ejecutará en la plataforma se conecta al simulador. Se verifican interfaces, serialización, relojes y pérdida de mensajes. La aceptación exige que la frecuencia efectiva y los comandos coincidan con el diseño.

Nivel 4: Hardware-in-the-Loop. La computadora de a bordo ejecuta el controlador y se comunica con una planta simulada. Se mide tiempo real, carga de la unidad central de procesamiento (CPU), memoria, temperatura y latencia. Se inducen retrasos y paquetes perdidos. La aceptación exige que el modo seguro se active de forma correcta.

Nivel 5: Pruebas de actuadores o vehículo asegurado. Se comprueban sentidos, escalas y límites con el vehículo sin libertad completa. Se verifica el botón de emergencia y la autoridad del piloto.

Nivel 6: Vuelo real de baja energía. Se utilizan baja altura, velocidad reducida y área amplia. Se activa un componente nuevo por vez. La aceptación exige seguimiento básico, telemetría completa y capacidad de aborto.

Nivel 7: Misión completa. Se ejecutan trayectorias, obstáculos o percepción en las condiciones objetivo. Se realizan repeticiones y ensayos de fallo controlado.

Cada puerta debe estar escrita. Por ejemplo: “el percentil 99 del ciclo es menor que 80 % del período”; “no existe violación de distancia”; “el error RMSE es menor que el umbral definido”; “la pérdida de objetivo activa flotación en menos de determinado tiempo”. Los umbrales dependen de la misión, pero deben fijarse antes de observar el resultado final.

La inyección de fallos forma parte de la validación: retraso de estado, paquete perdido, dato congelado, reinicio del solver, reducción de batería, perturbación de viento, error de obstáculo y pérdida de objetivo. No todos los fallos se prueban en vuelo; muchos se validan en SIL o HIL.

Para mantener seguridad operacional, el controlador experimental no debe anular la capacidad del autopiloto de estabilizar. El piloto debe disponer de un modo manual o de retorno. Se definen límites geográficos y de altura. Cada ensayo comienza con una reunión breve y termina con una revisión de incidentes.

14.14 Métricas, repeticiones e incertidumbre

La selección de métricas debe preceder al experimento. Para seguimiento se utilizan RMSE por eje, error tridimensional, ISE, error máximo y percentiles. Para orientación se informa el error angular obtenido del cuaternión, no la resta directa de componentes. Para control se registran magnitud, variación, saturación y suavidad.

Para identificación, se evalúan predicción a un paso, predicción libre a varios pasos y residuo. Una buena predicción a un paso puede deteriorarse rápidamente cuando el modelo se propaga. El horizonte de evaluación debe relacionarse con el del controlador.

Para seguridad, se informa distancia mínima, número y duración de violaciones, valor mínimo de la CBF, holgura máxima y estados de aborto. Para servo visual se reporta porcentaje de visibilidad, margen al borde, pérdida de objetivo y tiempo de recuperación. Para tiempo real se presentan mediana, percentiles 90, 95, 99 y máximo, junto con frecuencia de incumplimiento.

El promedio no es suficiente. Los controladores de optimización pueden presentar tiempos ocasionales altos. Un máximo aislado puede deberse a una tarea del sistema operativo, pero sigue siendo relevante si causa pérdida de ciclo. Por ello, se registra también carga de CPU, temperatura y proceso que comparte el equipo.

Los ensayos reales deben repetirse. Tres repeticiones ofrecen una primera impresión, pero más repeticiones son preferibles cuando el entorno es variable. Se reporta dispersión y no se selecciona únicamente el mejor vuelo. Si un ensayo se excluye, se conserva y se explica el criterio definido de antemano.

Una comparación justa alterna el orden de los controladores y utiliza baterías con condiciones semejantes. Las condiciones ambientales se registran. Cuando no es posible controlar el viento, se presenta como covariable o se agrupan vuelos dentro de un intervalo comparable.

La incertidumbre del sensor y del modelo puede propagarse mediante simulaciones Monte Carlo. Se varían masa, retardo, ruido, posición del obstáculo y condición inicial. Las semillas se publican. El resultado se expresa como distribución de errores y tasa de éxito. Esto no reemplaza la prueba real, pero revela regiones frágiles.

El informe final debe incluir resultados negativos. Si una configuración no converge, se documenta la combinación de parámetros y el diagnóstico. Evitar estas configuraciones puede ser más útil para otro laboratorio que mostrar únicamente el caso exitoso.

14.15 Paquete mínimo de réplica

Un paquete de réplica debería permitir que una persona ajena al proyecto produzca al menos una figura y ejecute un escenario corto. Debe contener:

- un README con requisitos, instalación y comandos
- licencia y forma de citación
- manifiestos de experimentos
- modelos de plataforma y archivos de calibración
- trayectorias y obstáculos
- configuraciones de controladores
- código de identificación, control y análisis
- entorno de software fijado
- datos crudos o un subconjunto representativo
- datos procesados y script que los genera
- pruebas automáticas
- un escenario MIL o SIL de ejecución rápida
- scripts para métricas y figuras
- descripción del hardware y cableado
- protocolo de seguridad
- lista de resultados esperados con tolerancias

El README debe presentar una ruta de cinco a diez minutos para comprobar la instalación y una ruta completa para la reproducción. La ruta corta puede ejecutar un modelo con datos incluidos. La completa puede requerir Webots o hardware.

Las figuras se generan mediante un único comando. El script debe leer los datos y la configuración, no valores copiados dentro del archivo de gráfica. Las etiquetas, unidades y filtros se definen de

forma centralizada. Cuando una figura combina varios experimentos, conserva los identificadores de origen.

Los datos se publican con un diccionario de campos. Si existen restricciones de tamaño, se incluye un subconjunto y se explica cómo solicitar el conjunto completo. Los registros que contengan información sensible se anonimizan sin alterar las señales de control.

La siguiente lista puede utilizarse antes de liberar una versión:

1. ¿El repositorio se instala desde una máquina limpia?
2. ¿Las versiones están fijadas?
3. ¿Existe un ejemplo de ejecución corto?
4. ¿Los datos crudos permanecen inmutables?
5. ¿Cada resultado indica el manifiesto que lo generó?
6. ¿Las unidades están en los nombres o en un diccionario?
7. ¿Los marcos y signos están documentados?
8. ¿Se publican parámetros del modelo y del controlador?
9. ¿El preprocesamiento puede reconstruirse?
10. ¿La división entrenamiento-validación-prueba está definida?
11. ¿Se incluyen trayectorias y condiciones iniciales?
12. ¿Se registran estados del solver y tiempos?
13. ¿Existe un controlador o modo de respaldo?
14. ¿Se documentan límites de seguridad?
15. ¿Las gráficas se generan sin edición manual?
16. ¿Se conservan ensayos fallidos?
17. ¿Las exclusiones tienen criterio previo?
18. ¿Se informa hardware, firmware y carga útil?

19. ¿Se distingue frecuencia nominal de frecuencia medida?
20. ¿Otra persona del equipo pudo repetir el ejemplo?

14.16 Ruta práctica para replicar los seis estudios

Una réplica ordenada puede desarrollarse en siete entregables. El primero es una interfaz común que lea el estado y envíe velocidades. Debe funcionar con un simulador y con el SDK real sin cambiar el controlador. El segundo es un conjunto de vuelos de identificación acompañado de manifiestos y datos crudos. El tercero contiene modelos Euler-Lagrange y DMDc comparados con datos independientes. El cuarto entregable implementa los controladores: cinemático, dinámico y NMPC con una trayectoria común. El quinto añade actualización DMDc, ley adaptativa y observador, comparando variantes mediante una perturbación repetible. El sexto incorpora el camino, CLF, CBF y obstáculos. El séptimo integra la cámara y el campo de visión.

Este orden reduce dependencias. El servo visual no debería utilizarse para depurar al mismo tiempo la dinámica, la calibración y el solver. Primero se comprueba el control con estado global; luego se sustituye gradualmente la fuente de realimentación.

Una planificación razonable asigna una versión estable a cada entregable. Cada versión conserva datos, métricas y un escenario de regresión. Si una extensión deteriora un resultado anterior, el cambio se detecta antes del vuelo.

Para el estudio del hexacóptero, la reproducción comienza con la interfaz de velocidades y la identificación de la dinámica reducida. Se comparan Euler-Lagrange y DMDc; después, se incorpora el modelo seleccionado al NMPC; primero, sin obstáculos y, luego, en simulación con

obstáculos. La evitación en vuelo real solo se intenta cuando existe percepción y se ha validado la cadena completa.

Para la comparación de controladores, se fija una trayectoria Lissajous o una equivalente con varios niveles de velocidad. Se ajustan los métodos con un conjunto de desarrollo y se reservan ensayos para comparación. Se registran precisión y tiempo de cálculo.

Para DMDc en línea, el modelo offline proporciona la condición inicial. Se ejecuta la actualización en modo sombra, se vigila excitación y luego se habilita la compensación. El cambio dinámico debe ser conocido en HIL antes de utilizar perturbaciones no controladas.

Para el NMPC adaptativo con observador, se implementan las cuatro variantes y se usa el mismo escenario. Se verifica por separado la estimación y el seguimiento. La activación en hardware se realiza de manera gradual.

Para NMPCC, se publica el camino, su parametrización y los obstáculos. Se prueba, primero, la geometría; después, CLF; luego, CBF y, finalmente, la combinación. Los conflictos se documentan mediante holguras y restricciones activas.

Para servo visual, se publica la calibración, el objetivo y el mundo de simulación. Se compara IBVS clásico y MPC con exactamente la misma secuencia. La pérdida del objetivo debe formar parte del protocolo.

Idea clave. La réplica más confiable avanza desde interfaces y datos hacia algoritmos. Intentar reproducir directamente la figura final suele ocultar errores de marcos, tiempo, preprocesamiento y seguridad.

14.17 Resumen del capítulo

La reproducibilidad en robótica aérea exige documentar plataforma, software, datos, tiempo, modelos, controladores, restricciones y criterios de aceptación. Los datos crudos deben conservarse; el procesamiento debe automatizarse; y cada figura debe rastrearse hasta un manifiesto. La identificación requiere excitación, separación de conjuntos y validación a varios pasos. El NMPC necesita escalado, warm start, vigilancia del solver y una política de respaldo. La adaptación y los observadores se evalúan mediante ablaciones y perturbaciones conocidas. Las CLF y CBF se verifican con métricas de seguridad y el servo visual requiere calibración, confianza y tratamiento explícito de pérdida del objetivo. La escalera MIL-SIL-HIL-vuelo real permite aumentar complejidad sin perder trazabilidad ni seguridad.

14.18 Preguntas de repaso

1. ¿Qué diferencia existe entre reproducción computacional, reproducción experimental y transferencia?
2. ¿Por qué los datos crudos deben considerarse inmutables?
3. ¿Qué señales deben registrarse para identificar un modelo de velocidades?
4. ¿Por qué no conviene dividir aleatoriamente muestras consecutivas entre entrenamiento y prueba?
5. ¿Qué diagnósticos deben vigilarse antes de aceptar una actualización DMDc en línea?
6. ¿Cómo se garantiza una comparación justa entre un controlador base y un NMPC?
7. ¿Qué información del solver debe registrarse en tiempo real?
8. ¿Por qué una trayectoria segura no demuestra por sí sola que una CBF funcionó?

9. ¿Qué elementos deben publicarse para repetir un experimento de servo visual?
10. Diseñe una puerta de aceptación para pasar de HIL a vuelo real.

Conclusiones

La modelación matemática de un robot aéreo multirrotor no consiste en escribir la ecuación más completa posible. Consiste en construir una representación adecuada para una tarea, una plataforma y una frecuencia. En vehículos comerciales, las velocidades de maniobrabilidad constituyen una interfaz práctica que permite reunir la cinemática del punto de interés con una dinámica reducida de alto nivel.

La utilidad científica de esa representación depende también de que el proceso pueda reconstruirse. Un resultado reproducible conserva los datos crudos, los manifiestos, el entorno de software, las configuraciones, los diagnósticos del solver y los criterios de aceptación. La réplica no exige copiar ciegamente los parámetros de una plataforma, sino repetir el procedimiento de medición, identificación, ajuste y validación sobre el nuevo sistema.

La formulación física mediante Euler-Lagrange aporta interpretación y estructura. La identificación basada en DMDc aproxima la respuesta observada y puede superar al modelo paramétrico cuando existen fenómenos agrupados difíciles de medir. La actualización en línea permite seguir cambios, siempre que los datos sean informativos y que el algoritmo vigile condición numérica, ruido y plausibilidad.

Los controladores de realimentación siguen siendo fundamentales. Son rápidos, transparentes y adecuados como primera implementación y como política de respaldo. La compensación dinámica mejora

el seguimiento sin resolver un problema de optimización. El NMPC se justifica cuando la misión exige anticipar, manejar saturaciones o incorporar restricciones. Su desempeño depende de discretización, escalado, horizonte, solver y presupuesto temporal.

Cuando el modelo no explica completamente la planta, la adaptación y los observadores reconstruyen una entrada equivalente. Su combinación puede mejorar robustez, pero añade estados y ganancias que requieren validación. Cuando la misión exige seguir una geometría y evitar obstáculos, el NMPCC separa contorno, avance y progreso; las CLF aportan convergencia y las CBF definen seguridad. Cuando la realimentación proviene de una cámara, el MPC mantiene las características dentro del campo de visión y conecta percepción con dinámica.

El hilo común de toda la obra es el proceso: definir marcos, seleccionar variables, identificar, controlar, imponer seguridad, medir tiempo y validar progresivamente. Ningún algoritmo aislado produce autonomía. La autonomía emerge de una arquitectura en la que cada componente conoce sus hipótesis, límites y mecanismos de respaldo.

Glosario

acados. Biblioteca de código abierto para resolver problemas de control óptimo en tiempo real mediante transcripción y métodos numéricos eficientes.

actitud. Orientación de un vehículo respecto a un marco de referencia.

adaptación. Modificación en línea de parámetros o acciones para compensar cambios o incertidumbres.

camino. Curva geométrica sin una asignación temporal obligatoria.

CasADi. Entorno simbólico para diferenciación automática y formulación de problemas de optimización numérica.

CBF. *Control Barrier Function*; función utilizada para mantener el estado dentro de un conjunto seguro.

CLF. *Control Lyapunov Function*; función utilizada para imponer o favorecer convergencia hacia un equilibrio.

cuaternión. Representación de orientación con cuatro componentes y norma unitaria.

dinámica no modelada. Parte de la respuesta real que no está representada por el modelo nominal.

disparo múltiple. Transcripción del OCP en la que estados de varios nodos son variables y se imponen restricciones de continuidad.

DMD. *Dynamic Mode Decomposition*; método basado en datos para aproximar evolución dinámica.

DMDc. *DMD with control*; extensión que incorpora entradas de control.

FOV. *Field of View* o campo de visión de la cámara.

HIL. *Hardware-in-the-Loop*; validación con hardware real y planta simulada en tiempo real.

IBVS. *Image-Based Visual Servoing*; servo visual cuyo error se define en coordenadas de imagen.

ISE. Integral del cuadrado del error.

jacobiano. Matriz que relaciona velocidades entre espacios o variables.

mapa Log. Transformación que lleva un error de rotación o cuaternión a un espacio tangente vectorial.

MIL. *Model-in-the-Loop*; validación en la que planta y controlador se representan como modelos.

MPC. *Model Predictive Control*; control que optimiza acciones futuras sobre un horizonte.

NMPCC. *Nonlinear Model Predictive Contouring Control*; MPC no lineal orientado al seguimiento de caminos.

NMPC. MPC que utiliza un modelo no lineal.

observador. Sistema dinámico que estima variables no medidas a partir del modelo, entradas y mediciones.

OCP. *Optimal Control Problem*; problema de control óptimo.

PBVS. *Position-Based Visual Servoing*; servo visual basado en una pose tridimensional estimada.

pose. Posición y orientación de un cuerpo.

punto de interés. Punto del robot distinto o no del centro de masa que se desea controlar, como una cámara.

RMSE. Raíz del error cuadrático medio.

ROS. *Robot Operating System*; middleware para organizar procesos, mensajes y herramientas robóticas.

SDK. Kit de desarrollo que expone datos y comandos de una plataforma.

SIL. *Software-in-the-Loop*; validación del código de control con una planta simulada.

subactuación. Situación en la que existen menos entradas independientes que grados de libertad.

trayectoria. Referencia geométrica asociada a una ley temporal.

Referencias

Ames, A. D., Xu, X., Grizzle, J. W., y Tabuada, P. (2017). Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control, 62*(8), 3861-3876. https://doi.org/10.1109/TAC.2016.2638961

Chaumette, F., y Hutchinson, S. (2006). Visual servo control. Part I: Basic approaches. *IEEE Robotics & Automation Magazine, 13*(4), 82-90. https://doi.org/10.1109/MRA.2006.250573

Corke, P. (2017). *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* (2nd ed.). Springer.

Guevara, B. S., Recalde, L. F., Moya, V., Varela-Aldás, J., Gandolfo, D. C., y Toibero, J. M. (2024). Estimation of unmodeled dynamics: Nonlinear MPC and adaptive control law with momentum observer dynamic. *IEEE Access, 12*, 77121-77132. https://doi.org/10.1109/ACCESS.2024.3407684

Guevara, B. S., Recalde, L. F., Varela-Aldás, J., Gandolfo, D. C., y Toibero, J. M. (2023b). Quadcopters control using online Dynamic Mode Decomposition. *IFAC-PapersOnLine, 56*(3), 589-594. https://doi.org/10.1016/j.ifacol.2023.12.088

Guevara, B. S., Recalde, L. F., Varela-Aldás, J., Andaluz, V. H., Gandolfo, D. C., y Toibero, J. M. (2023a). A comparative study between NMPC and baseline feedback controllers for UAV trajectory tracking. *Drones, 7*(2), 144. https://doi.org/10.3390/drones7020144

Guevara, B. S., Varela-Aldás, J., Moya, V., Cardona, M., Gandolfo, D. C., y Toibero, J. M. (2025). Model predictive contouring control with barrier and Lyapunov functions for stable path-following in UAV systems. *IEEE Access, 13*, 109742-109750. https://doi.org/10.1109/ACCESS.2025.3582505

Proctor, J. L., Brunton, S. L., y Kutz, J. N. (2016). Dynamic Mode Decomposition with control. *SIAM Journal on Applied Dynamical Systems, 15*(1), 142-161. https://doi.org/10.1137/15M1013857

Quan, Q. (2017). *Introduction to Multicopter Design and Control*. Springer.

Rawlings, J. B., Mayne, D. Q., y Diehl, M. (2020). *Model Predictive Control: Theory, Computation, and Design* (2nd ed.). Nob Hill Publishing.

Recalde, L. F., Guevara, B. S., Carvajal, C. P., Andaluz, V. H., Varela-Aldás, J., y Gandolfo, D. C. (2022a). System identification and nonlinear model predictive control with collision avoidance applied in hexacopters UAVs. *Sensors, 22*(13), 4712. https://doi.org/10.3390/s22134712

Recalde, L. F., Varela, J., Guevara, B. S., Andaluz, V., Giménez, J., y Gandolfo, D. (2022b). Constrained visual servoing of quadrotors based on model predictive control. *IFAC-PapersOnLine, 55*(37), 353-360. https://doi.org/10.1016/j.ifacol.2022.11.209

Schmid, P. J. (2010). Dynamic Mode Decomposition of numerical and experimental data. *Journal of Fluid Mechanics, 656*, 5-28. https://doi.org/10.1017/S0022112010001217

Siciliano, B., Sciavicco, L., Villani, L., y Oriolo, G. (2009). *Robotics: Modelling, Planning and Control*. Springer.

Zhang, H., Rowley, C. W., Deem, E. A., y Cattafesta, L. N. (2019). Online Dynamic Mode Decomposition for time-varying systems. *SIAM Journal on Applied Dynamical Systems, 18*(3), 1586-1609. https://doi.org/10.1137/18M1192329

Anexos

Anexo A. Notación matemática

Símbolo	Definición	Unidad o espacio
$\{I\}$	Marco inercial.	-
$\{B\}$	Marco solidario al vehículo.	-
$\{C\}$	Marco de la cámara.	-
$\boldsymbol{\eta}$	Pose reducida $[x, y, z, \psi]^T$	m, rad
$\boldsymbol{\nu}$	Velocidades de maniobrabilidad.	m/s, rad/s
$\boldsymbol{\mu}$	Referencias enviadas al vehículo.	m/s, rad/s
$\mathbf{J}$	Jacobiano cinemático.	variable
$\mathbf{M}$	Matriz de masa e inercia reducida.	variable
$\mathbf{C}$	Matriz de Coriolis y términos centrípetos.	variable
$\mathbf{A}, \mathbf{B}$	Matrices de un modelo DMDc.	variable
$\mathbf{x}$	Estado vectorial del controlador; distinto de las coordenadas escalares x, y, z.	$\mathbb{R}^n$
$\boldsymbol{\mu}$	Vector de referencias o entrada del controlador; distinto del píxel horizontal u.	$\mathbb{R}^m$
N	Número de pasos de predicción.	entero
Δt	Período de muestreo.	s
V	Función de Lyapunov.	escalar
h	Función de barrera.	escalar
$\mathbf{s}_v$	Vector de características visuales; distinto de la señal filtrada de adaptación.	píxeles o normalizado

Símbolo	Definición	Unidad o espacio
$\mathbf{q}$	Cuaternión unitario.	$\mathbb{H}$
ϑ	Parámetro de progreso del camino; distinto del ángulo de cabeceo.	m o adimensional
$\mathbf{e}_c$	Error de contorno.	m
$\mathbf{e}_l$	Error de avance.	m
s_f	Señal de error filtrada usada por la ley adaptativa.	variable

A.1 Convenciones

Los vectores se escriben en negrita minúscula y las matrices en negrita mayúscula. La notación $\| \mathbf{x} \|_{\mathbf{Q}}^2 = \mathbf{x}^T \mathbf{Q} \mathbf{x}$ representa una norma cuadrática ponderada. El símbolo † indica pseudoinversa. Las variables con tilde representan error y las variables con sombrero, estimación. La tipografía y los subíndices también separan símbolos que podrían confundirse entre capítulos.

A.2 Desambiguación de símbolos

El estado vectorial del controlador se distingue de las coordenadas cartesianas escalares; la referencia de control se separa de los píxeles de imagen; y las señales visuales, de las señales de adaptación. El parámetro de camino usa una variante de *theta* distinta del ángulo de cabeceo.

$$\mathbf{x} \neq x, \qquad \boldsymbol{\mu} \neq u, \qquad \mathbf{s}_v \neq s_f, \qquad \mathbf{q} \neq \mathbf{q}_{EL}, \qquad \vartheta \neq \theta$$

Anexo B. Parámetros y resultados reportados

B.1 Identificación del hexacóptero

Eje	ISE Euler-Lagrange	ISE DMDc	Interpretación
Longitudinal	1.26	0.56	DMDc reduce de forma importante el error
Lateral	0.35	0.16	DMDc reproduce mejor la respuesta
Vertical	0.0104	0.0100	Desempeño muy similar
Guiñada	0.104	0.038	DMDc presenta menor error

B.2 Configuraciones temporales representativas

Estudio	Tasa o período	Horizonte	Plataforma o entorno
Hexacóptero con NMPC	$\Delta t = 0.1$ s	2 s reportados en experimento	DJI Matrice 600 Pro
Comparación de controladores	Implementación en tiempo real	Según formulación NMPC	DJI Matrice 100
DMDc en línea	30 Hz	Ventana de 6 muestras	HIL DJI Matrice 100
NMPC adaptativo con MOD	30 Hz	30 pasos, 1 s	Webots/gemelo digital Matrice 100
NMPCC con CLF y CBF	30 Hz	30 pasos, 1 s	DJI Matrice 100, Jetson Orin NX
Servo visual predictivo	$\Delta t = 0.1$ s	10 pasos, 1 s	Webots, Mavic Pro 2 simulado

B.3 Resultados de tiempo de cálculo

Estudio	Resultado reportado	Observación
Comparación, simulación	0.054 ms cinemático; 0.141 ms dinámico; 17.54 ms NMPC	Mismo problema de seguimiento
Comparación, vuelo real	0.075 ms cinemático; 0.26 ms dinámico; 25.36 ms NMPC	NMPC fue más preciso
Hexacóptero NMPC	Menos de 100 ms	Compatible con período de 0.1 s
NMPC adaptativo con MOD	Menos de 5.2 ms en el entorno reportado	Horizonte de un segundo

B.4 Límites de interpretación

Los números de este anexo pertenecen a plataformas, configuraciones y equipos específicos. No deben utilizarse como garantías para otra implementación. Su valor consiste en ofrecer órdenes de magnitud y puntos de comparación. Toda nueva plataforma requiere identificar, medir tiempo y validar nuevamente.

Declaración sobre el uso de inteligencia artificial. Durante la elaboración de esta obra se utilizaron herramientas de inteligencia artificial (Chat GPT Versión 5.6 Sol Alta) como apoyo para organizar contenidos, revisar claridad expositiva y proponer ejemplos de implementación. Las formulaciones matemáticas, la interpretación de resultados, las referencias y la versión final fueron verificadas y aprobadas por el autor, quien asume plena responsabilidad por el contenido.

La modelación y el control de vehículos aéreos multirrotor constituyen una puerta de entrada al campo de la robótica aérea. Comprender cómo vuela un robot, predecir su comportamiento y lograr que siga una trayectoria de manera autónoma permiten integrar conocimientos de matemáticas, dinámica, control, percepción y programación.

Este libro propone un recorrido progresivo que comienza con la modelación cinemática, basada en la geometría y el movimiento del vehículo, y avanza hacia modelos dinámicos más completos mediante formulaciones como la de Euler-Lagrange. La identificación de sistemas complementa este proceso mediante el uso de datos experimentales para obtener modelos capaces de representar con mayor precisión la respuesta real del multirrotor.

En el ámbito del control, se abordan tanto la cinemática inversa, que transforma el movimiento deseado en referencias para el vehículo, como las estrategias que incorporan su dinámica y permiten analizar formalmente la estabilidad mediante herramientas como las funciones de Lyapunov.

El recorrido culmina con el control predictivo basado en modelo, una técnica conocida desde hace décadas que, gracias al aumento de la capacidad computacional, hoy puede implementarse en vehículos aéreos en tiempo real. Su principio consiste en anticipar el comportamiento futuro del sistema dentro de un horizonte temporal, optimizar las acciones de control y considerar simultáneamente el cumplimiento de restricciones, el seguimiento de trayectorias y la evitación de obstáculos.

Finalmente, la obra conecta la teoría con la práctica mediante la integración del vehículo, los sensores de percepción, la estimación del estado y las herramientas modernas de programación. De este modo, muestra cómo transformar los modelos y algoritmos en sistemas robóticos aéreos capaces de percibir, predecir y actuar de forma autónoma.

ISBN 978-9-942693-24-2

9 789942 693242