CEA Comité Español de Automática
XLV JORNADAS AUTOMÁTICA MÁLAGA 2024
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

Optimización de trayectorias y estabilización LQR para robot aéreo omnidireccional

del Río, J[a,*], Iriarte, I[a], Vilchez, L[a], Lasa, J[a], Lazkano, E[b], Rodriguez, I[b]

[a] *Tecnalia, Basque Research and Technology Alliance (BRTA), Mikeletegi 7 Pasealekua, 20009, San Sebastián, España.*
[b] *Robótica y Sistemas Autónomos (RSAIT), Universidad del País Vasco (UPV-EHU), Manuel Lardizabal 1, 20018, San Sebastián, España.*

To cite this article: del Río, J, Iriarte, I, Vilchez, L, Lasa, J, Lazkano, E, Rodriguez, I. 2024. Trajectory optimization and LQR-based stabilization for an omnidirectional aerial robot.
Jornadas de Automática, 45. https://doi.org/10.17979/ja-cea.2024.45.10816

Resumen

En este trabajo, se aborda el desarrollo de la planificación de trayectorias para un robot aéreo omnidireccional. La arquitectura del dron consiste en 4 quadrotores unidos con juntas omnidireccionales a un cuerpo central, permitiendo al sistema rotar 360º en los tres ejes mientras los quadrotores mantienen el sistema estable. Al tratarse de un sistema sobreactuado puede llegar de una posición o estado A a uno B por múltiples vías. Por ello, de las varias rutas posibles, es importante generar las que se ajusten a criterios de optimalidad, y así reducir el consumo del sistema. En el presente artículo se presenta una solución para generar trayectorias que se ajusten a ciertos criterios de optimalidad y restricciones del sistema. El problema se resuelve mediante el método de optimización de trayectorias de colocación directa, y posteriormente se utiliza la trayectoria generada como entrada en un lazo de control con estabilización LQR de tiempo finito. El trabajo se ha validado en simulación.

Palabras clave: Planificación de Trayectorias, Control no-lineal y optimización, Control con restricciones, Estabilidad de sistemas no lineales, Problemas de gran escala de control óptimo, Robótica aérea.

Trajectory optimization and LQR-based stabilization for an omnidirectional aerial robot

Abstract

In this work, we address the path planning for an omnidirectional aerial robot. The drone's architecture consists of 4 quadrotors connected with omnidirectional joints to a central body, allowing the system to rotate 360º in all three axes while the quadrotors maintain stability. As an overactuated system, it can reach from position or state A to B through multiple rutes. Therefore, of the various possible routes, it is important to generate those that meet optimality criteria, thereby reducing system consumption. In this article, we propose a solution to generate trajectories that meet certain optimality criteria and satisfy system constraints. The problem is solved using the direct collocation optimization method, and the generated trajectory is subsequently used as input to a control loop that stabilizes along the trajectory using a finite-time LQR controller. The work was validated by simulations.

Keywords: Trajectory and Path Planning, Nonlinear control and optimization, Constrained control, Stability of nonlinear systems, Large-scale optimal control problems, Flying robots.

1. Introducción

En los últimos años, se ha producido un gran avance tecnológico en diversas ramas de la ingeniería, incluyendo el desarrollo de vehículos aéreos. En contraposición a soluciones más convencionales (p.e helicópteros), los vehículos multirrotor están ganando popularidad. Además de un menor coste, consumo de energía, generación de ruido y emisiones, ofrecen una mayor maniobrabilidad, haciéndolos especialmente adecuados para operaciones de búsqueda y rescate, inspección, vigilancia y filmación entre otras (p.e Pitas (2021), Gonçalves

*Autor para correspondencia: josdriuu@gmail.com

et al. (2022)).

Estos vehículos multirrotor son mayoritariamente de ejes paralelos, es decir, llevan todos los rotores alineados en la misma dirección. Así, son vehículos subactuados, con un número de grados de libertad controlables mayor que el número de actuadores controlados. Esta limitación dificulta su maniobrabilidad en entornos complejos, y también en tareas de manipulación. Algunas propuestas de manipulación consisten en acoplar otros sistemas como brazos robóticos (p.e Szász et al. (2022) o Ramalepa and Jamisola Jr (2021)).

Para suplir esas carencias existen propuestas de arquitecturas con omnidireccionalidad parcial o incluso completa (Park et al. (2016)). Estas propuestas suelen otorgar omnidireccionalidad al sistema con actuadores que generan esfuerzos internos, agregando peso y movimientos que no transmiten fuerza de propulsión. El dron omnidireccional (Figura 1) es una variación del concepto propuesto en Iriarte et al. (2020), y pretende solucionar ese inconveniente. Cuenta con 4 quadrotores unidos a una estructura central a través de juntas omnidireccionales pasivas. De esta forma son las hélices las que generan las rotaciones en las juntas, permitiendo posicionar el vehículo en cualquier dirección, y a su vez estabilizarlo.

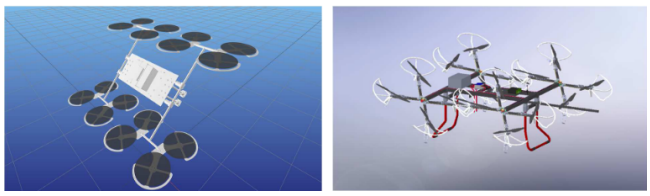

Figura 1: Prototipo del robot aéreo omnidireccional diseñado por Tecnalia.

El objetivo es desarrollar un planificador de trayectorias factibles para este vehículo, teniendo en cuenta que su omnidireccionalidad permite tareas que exijan traslaciones y/o rotaciones, y que las trayectorias puedan estar sujetas a restricciones y deban cumplir ciertos requisitos de optimalidad.

El artículo se estructura de la siguiente forma. La sección 2 describe los trabajos relacionados (estado del arte). La sección 3 presenta la dinámica del robot aéreo para entender el contexto de trabajo. La sección 4 presenta el método de planificación de trayectorias por colocación directa propuesto y la estabilización de la misma mediante un LQR de horizonte finito. Las simulaciones realizadas para verificar la validez de la propuesta se describen en la sección 5, seguida por la sección 6 en el que se muestran los resultados obtenidos. Finalmente, el artículo concluye con una última sección dedicada a las conclusiones y el trabajo futuro.

2. Revisión del estado del arte

La planificación de trayectorias en robótica se encarga de determinar la ruta que debe seguir un robot, en la mayoría de casos siguiendo algún criterio de optimalidad, desde su posición actual hasta un objetivo específico, evitando obstáculos y respetando limitaciones cinemáticas y dinámicas.

Esta planificación implica resolver una serie de problemáticas. En primer lugar, el espacio de configuración del robot, que representa todas las posibles posiciones y orientaciones que el robot puede alcanzar, puede ser extremadamente grande y complicado, especialmente en entornos con obstáculos o en robots sobreactuados como el robot omnidireccional. Por lo tanto, encontrar una ruta factible y óptima en este espacio es un problema no trivial, para el que actualmente existen distintas soluciones (Gasparetto et al. (2015)). Existen múltiples propuestas de aplicación de planificación de trayectorias con diferentes métodos y arquitecturas (Gugan and Haque (2023), Saeed et al. (2022)).

Los métodos de planificación de trayectorias basados en optimización son muy interesantes, ya sean tradicionales como el usado en Kong et al. (2023), o propuestas más avanzadas para disminuir el coste computacional como en Kondo et al. (2024).

La principal contribución de este artículo consiste en la aplicación del método de colocación directa descrito en Tedrake (2023) al sistema novedoso que es el robot aéreo omnidireccional. Se ha escogido la colocación directa porque tiene la particularidad de permitir formular y resolver el problema de optimización con un pequeño número de funciones de evaluación, variables de decisión y restricciones, lo que resulta en soluciones más precisas y eficientes.

Así, se propone resolver los movimientos del sistema pudiendo imponer restricciones en el problema de optimización, pretendiendo conseguir que de las múltiples rutas que pueda tomar gracias a la omnidireccionalidad, el dron realice la óptima sujeto a las restricciones y a la función de coste que penalizará los estados indeseados. Además, dichas trayectorias se estabilizarán con un LQR de tiempo finito, a diferencia del sistema LQR aplicado anteriormente en Iriarte et al. (2021).

3. Dinámica del sistema

Un sistema se denomina sobreactuado cuando tiene más acciones de control que el número de grados de libertad (DOF) a controlar. El robot aéreo omnidireccional cuenta con 14 grados de libertad. Al igual que un dron convencional, posee 6 grados de libertad (3 traslaciones y 3 rotaciones) que hacen referencia al sólido rígido central. El resto de los grados de libertad hacen referencia a las rotaciones que los quadrotores pueden realizar respecto al cuerpo central.

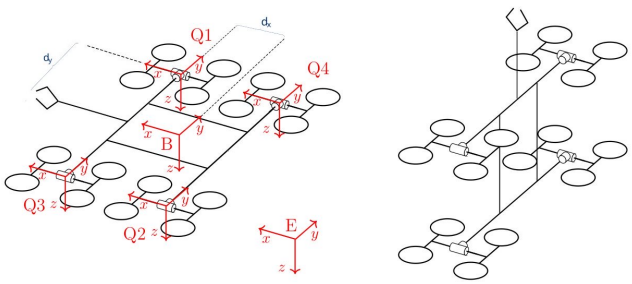

Figura 2: *Representación del robot aéreo omnidireccional con los 6 sistemas de referencia. E representa el sistema de referencia Earth, B el sistema de referencia del cuerpo y cada uno de los Qi, el sistema de referencia del i-ésimo quadrotor.*

Concretamente, cada i-ésima junta omnidireccional puede rotar en α_i y β_i (Roll y Pitch) respecto al cuerpo central. Sumando los cuatro quadrotores, quedan un total de 14 grados de

libertad. En la Figura 2 se representa un diagrama del sistema, además de una rotación del cuerpo central en "pitch" con los quadrotores estabilizados.

La arquitectura permite posicionar el cuerpo central en la dirección deseada, mientras los quadrotores se encargan de ejecutar las fuerzas y pares, y la estabilización.

Los vectores de acciones de control (**u**) y estados (**x**) quedan definidos de la siguiente forma:

$$\mathbf{u} = (u_1, ..., u_{16}) \quad (1)$$

$$\mathbf{x} = (q, r, \alpha_1, \beta_1, ..., \alpha_4, \beta_4, w, v, w_{\alpha_1}, w_{\beta_1}, ..., w_{\alpha_4}, w_{\beta_4}), \quad (2)$$

donde:

- u_i [N]: Fuerza de la i-ésima hélice.
- q: Vector de rotaciones en cuaterniones del cuerpo central (q_w, q_x, q_y, q_z).
- r [m]: Vector de traslaciones del cuerpo central (r_x, r_y, r_z).
- α_i, β_i [º]: Ángulos de rotación del i-ésimo quadrotor.
- w [rad/s]: Vector de velocidades angulares del cuerpo central(w_x, w_y, w_z).
- v [m/s]: Vector de velocidades lineales del cuerpo central(v_x, v_y, v_z).
- $w_{\alpha_i}, w_{\beta_i}$ [rad/s]: Velocidades angulares del i-ésimo quadrotor.

4. Planificación de trayectorias

4.1. Optimización de Trayectorias

El problema de los métodos de programación dinámica es que muchas de estas soluciones sirven para sistemas con pocos grados de libertad, pero no escalan adecuadamente a sistemas de más de cuatro o cinco dimensiones. Generalmente, se linealiza alrededor de un punto de operación nominal, siendo comúnmente el equilibrio del sistema (por ejemplo, con un regulador LQR). Aunque este método es útil incluso para sistemas de alta dimensionalidad, el control está limitado a la región del espacio de estados donde la linealización es una buena aproximación de la dinámica no lineal.

Sea el modelo $\dot{x} = f(x, u)$, con una condición inicial x_0 y una trayectoria $u(t)$ definida en un intervalo finito de tiempo $t \in [t_0, t_f]$. Se define como función de coste de intervalo finito:

$$J_{u(\cdot)}(x_0) = l_f(x(t_f)) + \int_{t_0}^{t_f} l(x(t), u(t))\, dt, \quad (3)$$

siendo l_f un coste final, que se sumará a la ecuación para dar más importancia al estado final. Entonces, definimos el problema de optimización como:

$$\begin{aligned} \min_{u(\cdot)} \quad & J_{u(\cdot)}(x_0) \\ \text{sujeto a} \quad & \dot{x}(t) = f(x(t), u(t)), \forall t \in [t_0, t_f] \\ & x(t_0) = x_0. \end{aligned} \quad (4)$$

Una vez definido el sistema, se pueden agregar diferentes restricciones para adecuar el sistema, como evitación de obstáculos (lugares en el espacio que el robot no debe alcanzar) o limitaciones de la dinámica del sistema a optimizar, como limitaciones en las entradas (p.e $u_{min} \leq u \leq u_{max}$).

4.2. Colocación Directa

En la colocación directa, la entrada y los estados son representados como funciones polinomiales por partes. En concreto, el algoritmo se caracteriza por emplear como aproximación de la $u(t)$ un polinomio de primer orden y $x(t)$ como un polinomio cúbico.

Entonces, las únicas variables de decisión que necesitaremos en nuestro optimizador serán los valores de $u(t)$ y $x(t)$ en los "puntos de ruptura" o "break points" (instantes en los que se toman decisiones sobre las entradas y estados del sistema). Para la interpolación de primer orden en $u(t)$, dados $u(t_k)$ y $u(t_{k+1})$ se puede resolver para todos los valores de $u(t)$ entre el intervalo de $t \in [k, k+1]$. Además, teniendo $x(t_k)$ y $u(t_k)$, se puede resolver $\dot{x}(t_k) = f(x(t_k), u(t_k))$.

Con la dinámica del sistema y los valores de las variables de decisión, la curva o el "spline" está completamente especificada. Quedaría definir restricciones adicionales para que $x(t_{k+1})$ sea aproximadamente $x(t_k) + \int_{t_k}^{t_{k+1}} f(x(t), u(t))\, dt$. En otras palabras, se busca una transición suave y coherente entre los estados del sistema a lo largo del tiempo. En la colocación directa esto se define con restricciones derivativas en los puntos de ruptura.

Finalmente, el sistema quedaría representado de la siguiente forma:

$$\begin{aligned} \min_{x[.],u[.]} \quad & l_f(x(N)) + \sum_{n=0}^{N-1} h_n l(x[n], u[n]) \\ \text{sujeto a} \quad & \dot{x}(t_{c,n}) = f(x(t_{c,n}), u(t_{c,n})), \forall n \in [0, N-1] \\ & x[0] = x_0, \end{aligned} \quad (5)$$

siendo $x[k] = x(t_k)$ las variables de decisión y $t_{c,k}$ el punto donde la restricción de colocación depende de las variables de decisión: $x[k]$, $x[k+1]$, $u[k]$, $u[k+1]$. Y h_n el el tamaño del paso de tiempo asociado con el intervalo n en la discretización del horizonte temporal.

4.3. Trayectoria Inicial

Antes de aplicar el método de optimización, hay que generar una trayectoria inicial a optimizar. Esta será una interpolación lineal entre los estados inicial $x(t_0)$ y final $x(t_f)$, en caso de haber más de dos estados se realiza una interpolación en cada intervalo entre dos estados y se concatenan.

También se genera la serie de valores de entrada de los propulsores a lo largo del tiempo. En este caso se inicia la optimización con una entrada u_0 estable en el tiempo que contrarreste la fuerza de la gravedad.

$$u_0 = \frac{m \cdot g}{N_m}, \quad (6)$$

siendo:

- m [kg]: Peso total del sistema.
- g [m/s²]: Constante gravitatoria.
- N_m: Número de motores.

4.4. Función de coste para el robot aéreo omnidireccional

Para generar las trayectorias, se debe definir una función de coste a minimizar que haga referencia a los vectores de entrada (Eq. 1) y estados (Eq. 2), y que penalice adecuadamente aquellas variables que no interesa que varíen a lo largo del tiempo. Las matrices de coste son las que almacenan los parámetros de ponderación para la penalización. Se escoge una función de coste cuadrática para poder emplearla tanto en el optimizador de trayectorias como en el LQR.

Determinar un conjunto de coeficientes para las matrices Q y R no es una tarea trivial, como se explica en Iriarte et al. (2021). El número de parámetros libres para ajustar en este caso suma un total de 571. Estos parámetros deben proporcionar información sobre la importancia relativa de cada variable de estado y de control en el problema de control general, teniendo en cuenta las unidades en las que se expresa cada magnitud. Dependiendo de los coeficientes elegidos, el rendimiento puede variar significativamente. Por lo tanto, es necesario encontrar un conjunto de coeficientes razonables que generalicen los movimientos de la arquitectura del vehículo. Cuando no existe una penalización particularmente fuerte asociada con tener simultáneamente errores en una combinación de estados o con utilizar simultáneamente un conjunto de actuadores, es práctica común despreciar los elementos no diagonales de ambas matrices, lo que reduce el conjunto de parámetros libres a 45 (Matrices 7 y 8).

Matriz $Q \in \mathbb{R}^{29\times29}$:

$$Q = \begin{bmatrix} k_q & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & k_w \end{bmatrix}. \tag{7}$$

Matriz $R \in \mathbb{R}^{16\times16}$:

$$R = \begin{bmatrix} k_{u_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & k_{u_{16}} \end{bmatrix}. \tag{8}$$

Siendo k_i las constantes que multiplicarán los estados para penalizarlos.

Un reto que supone la omnidireccionalidad es el de establecer unos parámetros que generalicen todos los casos de movimiento.

4.5. Restricciones

La omnidireccionalidad le permite al robot aéreo llegar de un estado a otro por múltiples trayectorias. Por un lado, la función de coste limitará varias de las rutas definiendo movimientos óptimos para nuestro caso específico. Pero las restricciones también son una parte esencial para definir aquellos movimientos realizables o para limitar la región de espacio de operación (obstáculos).

Las restricciones permiten definir, por un lado, características del modelo dinámico (p.e fuerzas mínimas y máximas de los propulsores o ángulos de giro que pueden realizar las juntas omnidireccionales en α_i y β_i).

Además, también se pueden definir ciertas restricciones específicas para cada trayectoria. Por ejemplo, si se quieren evitar ciertos movimientos como la rotación en z mediante diferencia de par. Es conveniente evitar esto, ya que es energéticamente ineficiente. Las restricciones tendría la siguiente forma:

$$u_i + u_{i+1} - u_{i+2} - u_{i+3} = 0, \tag{9}$$

para $i = (0, 4, 8, 12)$.

Dicho de otro modo, el momento vertical en cada quadrotor será 0, anulándose las hélices que rotan en sentido horario con las que rotan en antihorario (fuerzas horarias equivalentes a antihorarias).

E incluso se pueden definir restricciones para evitar ciertas regiones del espacio, por ejemplo, si se quieren evitar obstáculos o mantener relaciones específicas con los mismos.

4.6. Lazo de control

La trayectoria se utiliza como entrada para el lazo de control, posteriormente se realiza la estabilización con un regulador LQR de tiempo finito (Figura 3).

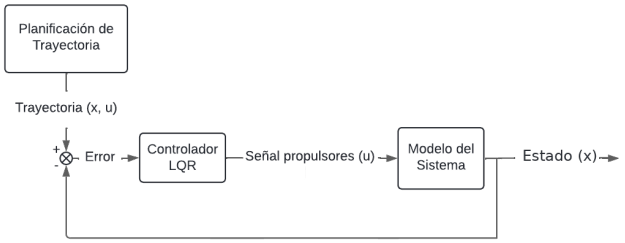

Figura 3: Diagrama del lazo de control con entrada de trayectoria.

El LQR de horizonte finito se utiliza cuando se necesita calcular un control óptimo para un horizonte de tiempo limitado, mientras que el LQR de tiempo infinito busca una ley de control óptima indefinida (Rodrigo and Patiño (2015)). El estabilizador de horizonte finito es esencial para estabilizar el sistema a lo largo de una trayectoria, ya que la colocación directa construye la trayectoria mediante puntos de colocación, aproximándolos por parábolas. El estabilizador asegura que el sistema llegue de un punto de colocación a otro sin desviarse.

5. Simulaciones

El funcionamiento se valida definiendo un entorno de simulación con librería Drake, específica para sistemas dinámicos (Tedrake (2019)). El entorno de simulación se ejecuta en una instancia que proporciona una visualización 3D de la actuación del sistema (Figura 4).

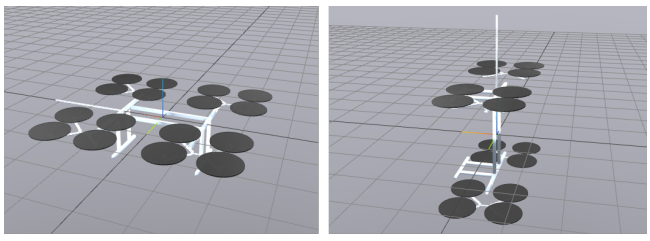

Figura 4: *Instancia de simulación con el modelo del robot aéreo omnidireccional. A la izq. el sistema está en posición horizontal, a la dcha. rotado -90º en Pitch.*

En la Tabla 1 se recogen los parámetros físicos definidos en el URDF del robot aéreo omnidireccional.

Tabla 1: Parámetros del robot aéreo omnidireccional.

Variable	**Valores**			**Unidad**
l_prop	0.629			kg·m^2
m_b	18			kg
m_q	1.5			kg
m_p	0.017			kg
	x	y	z	
CQ_i	0.57	0.66	0	m
QP_i	0.57	0.66	0	m
Iii_b	1.765e-17	6.297e-17	7.412e-17	kg·m^2
Iii_q	0.069	0.075	0.143	kg·m^2
Iii_p	1.26e-15	1.26e-15	1.264e-15	kg·m^2

Siendo:

- CQ_i: Valores que representan las coordenadas del centro de masa en cada i-ésimo eje.
- QP_i: Valores que representan el centro de presión en cada i-ésimo eje.
- l_prop: Valores que representan las coordenadas del centro de masa en cada eje.
- m_b, m_q, m_p: Masas de diferentes componentes del dron. Cuerpo principal, quadrotores y hélices respectivamente.
- Iii_b: Momentos de inercia del cuerpo principal alrededor del i-ésimo eje.
- Iii_q: Momentos de inercia de los quadrotores alrededor del i-ésimo eje.
- Iii_p: Momentos de inercia de las hélices alrededor del i-ésimo eje.

De las numerosas simulaciones realizadas, se presentan las dos de mayor relevancia. La primera pone en práctica la restricción de la Eq. 9. La segunda refleja un caso más práctico, simulando una inspección de tubería, limitando la región del espacio de la misma con restricciones, pero acercando el palpador (extensión del cuerpo central donde colocar la herramienta) para la inspección.

Se debe escoger una función de coste que generalice los casos de uso. Esta se utiliza para la optimización de las trayectorias, así como para la posterior estabilización LQR de tiempo finito. Se penalizan los estados relacionados con las rotaciones en Roll y Pitch con el fin de mantener el cuerpo central estable en el plano xy cuando no sea imprescindible que realice dichas rotaciones. Para la matriz R todos los parámetros k_{ui} tendrán un valor de 1. Los valores de la matriz Q se recogen en la Tabla 2.

Tabla 2: Parámetros de la matriz Q.

	w	x	y	z	α_i	β_i	Unidad
K_{qi}	1	100	100	0	-	-	Cuaternión
K_{ri}	-	1	1	1	-	-	m
$K_{\alpha\beta}$	-	-	-	-	1	1	rad
K_{wi}	-	100	100	1	-	-	rad/s
K_{vi}	-	1	1	1	-	-	m/s
$K_{w\alpha w\beta}$	-	-	-	-	1	1	rad/s

Las simulaciones se han realizado en un equipo con un procesador *13th Gen Intel® Core™ i9-13900K×32*. Los tiempos de ejecución se miden para el flujo de programa completo, estas pueden variar en las simulaciones realizadas entre un rango de 1 a 10 minutos. Esto ocurre en función de lo compleja que sea la trayectoria a realizar.

6. Resultados

Los resultados de las dos simulaciones se reflejan mediante gráficas donde se observa la trayectoria generada para cada estado a lo largo del tiempo, superpuesta con la estabilización realizada por el controlador. Para simplificar, de los 29 estados, se presentan únicamente los estados de rotaciones en cuaterniones y traslaciones del cuerpo central.

6.1. *Simulación con restricción de par en z para rotar el dron en Yaw*

El objetivo es rotar el sistema, manteniendo la estabilidad en el plano xy y evitando diferencias de par en las hélices. Para ello, además de la restricción de la Eq. 9 en el optimizador, se busca una trayectoria que mantenga el cuerpo central estable en el plano xy. Se penalizan los giros en xy (Tabla 2).

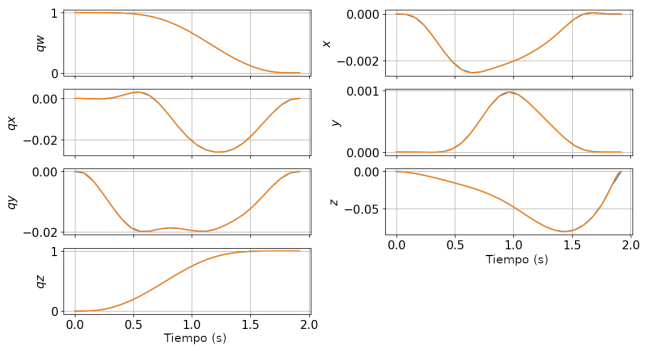

Figura 5: *Valores de los estados (rotaciones y traslaciones) a lo largo del tiempo en el giro de 180º.*

Para validar que los giros se realizan correctamente, se han realizado simulaciones de giro de 45º, 90º y 180º. En la Figura 5 se muestra la simulación correspondiente al giro de 180º.

El cambio en las rotaciones en los ejes x e y oscila entre los valores 0 y -0.02 aproximadamente por cuaternión, o lo que es lo mismo, una rotación máxima de 1.8º en x e y si se pasa a ángulos de Euler. Se considera que es una variación asumible para realizar el giro en z.

El tiempo de ejecución para esta simulación ha sido de 1'52.6". La simulación se puede observar en el siguiente vídeo.

6.2. *Simulación de inspección de tubería*

El caso práctico de inspección supone realizar un giro que mantenga el palpador del sistema tangente al centro de la tubería. De esta forma, el robot aéreo realiza una rotación en el eje y, mientras se traslada en x y z, cubriendo 45º de la tubería.

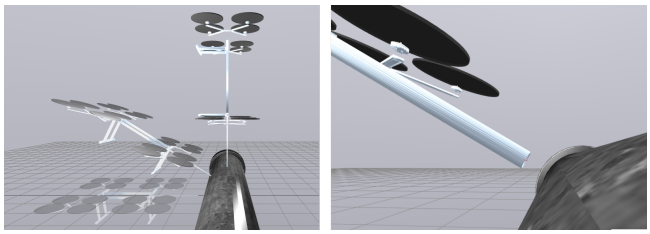

Figura 6: Simulación de inspección de tubería.

Para evitar que el robot aéreo colisione con la tubería se define una restricción de espacio. Se utiliza la ecuación de la circunferencia en el plano xz, ya que la tubería será tangente al mismo:

$$(x - x_t)^2 + (z - z_t)^2 >= r_t^2, \tag{10}$$

siendo:

- x_t: Posición en x del centro de la tubería.
- z_t: Posición en z del centro de la tubería.
- r_t: Radio la tubería.

De esta forma se realizará la trayectoria entre los estados definidos, evitando la región del espacio de la circunferencia del plano xz, o lo que es lo mismo, de un cilindro.

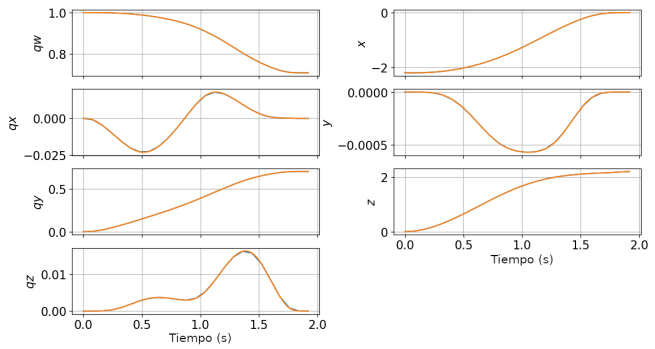

Figura 7: *Valores de los estados (rotaciones y traslaciones) a lo largo del tiempo en la inspección de tubería.*

El tiempo de ejecución correspondiente a esta simulación es de 3'9" (vídeo).

7. Conclusiones

Las simulaciones muestran resultados prometedores, aunque hay aspectos por mejorar. Es necesario equilibrar restricciones y función de coste para adaptarse a trayectorias más complejas y mejorar el tiempo de cómputo.

La línea de trabajo debe enfocarse ahora en mejorar el sistema para adecuarse a todas las trayectorias y hacerlas más robustas. Además, una línea importante que seguir es la de reducir el tiempo de cómputo para poder acercarse cada vez más al tiempo real. Las propuestas principales existentes siguen dos líneas. Por un lado, buscar algún método más complejo para generar trayectorias iniciales similares a la factible, reduciendo la carga de trabajo del optimizador. La otra línea sería utilizar el trabajo realizado para entrenar una red neuronal de aprendizaje por refuerzo, reduciendo considerablemente el tiempo de cómputo de cada trayectoria. Estas propuestas se encuentran actualmente en planteamiento.

Otro aspecto a avanzar es introducir perturbaciones al sistema (p.e. viento) y ajustar la robustez del LQR para rechazarlas, acercando así la solución a la aplicación real.

Referencias

Gasparetto, A., Boscariol, P., Lanzutti, A., Vidoni, R., 2015. Path planning and trajectory planning algorithms: A general overview. Motion and Operation Planning of Robotic Systems: Background and Practical Approaches, 3–27.
DOI: 10.1007/978-3-319-14705-5

Gonçalves, R. S., Souza, F. C., Homma, R. Z., Sudbrack, D. E. T., Trautmann, P. V., Clasen, B. C., 2022. Robots for inspection and maintenance of power transmission lines. In: Robot Design: From Theory to Service Applications. Springer, pp. 119–142.
DOI: 10.1007/978-3-031-11128-0

Gugan, G., Haque, A., 2023. Path planning for autonomous drones: Challenges and future directions. Drones 7 (3), 169.
DOI: 10.3390/drones7030169

Iriarte, I., Iglesias, I., Lasa, J., Calvo-Soraluze, H., Sierra, B., 2021. Enhancing vtol multirotor performance with a passive rotor tilting mechanism. IEEE Access 9, 64368–64380.
DOI: 10.1109/ACCESS.2021.3075113

Iriarte, I., Otaola, E., Culla, D., Iglesias, I., Lasa, J., Sierra, B., 2020. Modeling and control of an overactuated aerial vehicle with four tiltable quadrotors attached by means of passive universal joints. In: 2020 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, pp. 1748–1756.
DOI: 10.1109/ICUAS48674.2020.9213848

Kondo, K., Tagliabue, A., Cai, X., Tewari, C., Garcia, O., Espitia-Alvarez, M., How, J. P., 2024. Cgd: Constraint-guided diffusion policies for uav trajectory planning. arXiv preprint arXiv:2405.01758.
DOI: 10.48550/arXiv.2405.01758

Kong, F., Li, J., Jiang, B., Wang, H., Song, H., 2023. Trajectory optimization for drone logistics delivery via attention-based pointer network. IEEE Transactions on Intelligent Transportation Systems 24 (4), 4519–4531.
DOI: 10.1109/TITS.2022.3168987

Park, S., Her, J., Kim, J., Lee, D., 2016. Design, modeling and control of omni-directional aerial robot. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1570–1575.
DOI: 10.1109/IROS.2016.7759254

Pitas, I., 2021. Drone vision and deep learning for infrastructure inspection. In: 2021 IEEE International Conference on Autonomous Systems (ICAS). IEEE.
DOI: 10.1109/ICAS49788.2021.9551136

Ramalepa, L. P., Jamisola Jr, R. S., 2021. A review on cooperative robotic arms with mobile or drones bases. International Journal of Automation and Computing 18 (4), 536–555.
DOI: 10.1007/s11633-021-1299-7

Rodrigo, R. H., Patiño, D. H., 2015. Numerical solution to uncertainties in the finite time optimal control systems design. In: 2015 XVI Workshop on Information Processing and Control (RPIC). IEEE, pp. 1–5.
DOI: 10.1109/RPIC.2015.7497169

Saeed, R. A., Omri, M., Abdel-Khalek, S., Ali, E. S., Alotaibi, M. F., 2022. Optimal path planning for drones based on swarm intelligence algorithm. Neural Computing and Applications 34 (12), 10133–10155.
DOI: 0.1007/s00521-022-06998-9

Szász, R., Allenspach, M., Han, M., Tognon, M., Katzschmann, R. K., 2022. Modeling and control of an omnidirectional micro aerial vehicle equipped with a soft robotic arm. In: 2022 IEEE 5th International Conference on Soft Robotics (RoboSoft). IEEE, pp. 01–08.
DOI: 10.1109/RoboSoft54090.2022.9762161

Tedrake, R., 2019. Model-based design and verification for robotics.
URL: https://drake.mit.edu/

Tedrake, R., 2023. Underactuated Robotics.
URL: https://underactuated.csail.mit.edu