UNIVERSIDAD DE GUANAJUATO

CAMPUS IRAPUATO - SALAMANCA
DIVISIÓN DE INGENIERÍAS

Sistema para la Ejecución de Trayectorias en Drones Aéreos

TESIS PROFESIONAL

QUE PARA OBTENER EL GRADO DE:

MAESTRO EN INGENIERÍA ELÉCTRICA
(Opción: Instrumentación y Sistemas Digitales)

PRESENTA:

Ricardo Enrique Lugo Ochoa

DIRECTOR:

Dr. Víctor Ayala Ramírez

Salamanca, Gto. Mayo 2018

Abstract

Los drones aéreos son vehículos aéreos no tripulados (UAVs) que han sido foco de interés en numerosos estudios por el gran potencial que poseen sobre diversas áreas. El más destacado en los últimos años es el cuadricóptero o cuadrirrotor, un helicóptero con cuatro rotores ubicados en formación cuadrada en torno al centro de masa, que se caracteriza por poseer una gran estabilidad contrastada con una mayor complejidad en su modelado y control. El estudio toma como base el AR.Drone 2.0 de la marca Parrot, una plataforma muy utilizada en investigación por brindar facilidad de conexión con un ordenador además de existir software ya desarrollado para su comunicación con el entorno ROS. En primera instancia se exponen algunos conceptos clave para la investigación y se explica el modo de funcionamiento del cuadricóptero. Después se deducen los modelos cinemático y dinámico que rigen su comportamiento, así como qué características presenta el ejemplar en estudio. Luego se procede a desarrollar e implementar el control del cuadricóptero con dos enfoques de interés, un controlador inteligente, que se sustenta en una técnica tomada de la inteligencia artificial, la lógica difusa, y un controlador convencional, ampliamente utilizado en la literatura para el seguimiento de trayectorias y en la mayoría de controles de la industria. En las pruebas, se emplean un conjunto de trayectorias definidas, siendo el error respecto a las mismas el criterio para medir el desempeño de los controladores. Finalmente se presentan los resultados obtenidos y las conclusiones más resaltantes del trabajo realizado.

Palabras claves: AR.Drone 2.0, ejecución de trayectoria, controlador difuso, controlador PID.

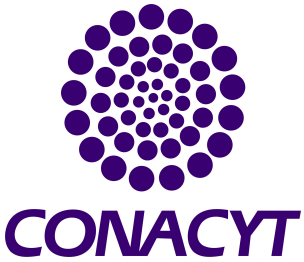
Agradecimientos

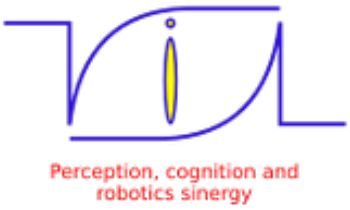
Un agradecimiento a CONACYT (Consejo Nacional de Ciencia y Tecnología), por el apoyo económico brindado durante la realización de la investigación, identificado por el registro (CVU/Becario) 736556/598648.

Gracias a la Universidad de Guanajuato, por fomentar el desarrollo de profesionales íntegros tanto en lo académico como en lo personal.

Asimismo a LaViRIA (Laboratorio de Visión, Robótica e Inteligencia Artificial), por ser el ambiente más apropiado para desarrollar habilidades profesionales.

Finalmente, un especial agradecimiento a todas las personas que de una u otra manera contribuyeron en el logro de esta meta.





Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En el primer capítulo se da a conocer el tema de investigación, describiendo el problema de interés y las motivaciones del proyecto. También se presenta el objetivo principal junto a los específicos que lo hacen posible. Finalmente se hace una revisión general del estado del arte y trabajos relacionados.

1.1 Descripción del Problema

Mucho del confort de la vida cotidiana es producto de la aplicación de los sistemas de control. Estos sistemas se encuentran presentes en diversos sectores, tales como la industria de la manufactura, dispositivos para el cuidado de la salud, sistemas financieros, entre otros. En virtud del continuo crecimiento en la complejidad de los sistemas, resulta de vital importancia el desarrollo de métodos más avanzados y efectivos para controlarlos.

Una aplicación en donde es imprescindible un control de esta índole es en la ejecución de trayectorias sobre drones aéreos. En especial, los de tipo cuadricóptero se han vuelto muy populares en los últimos años tanto en la investigación académica como en aplicaciones industriales [1, 2]. Existen incontables pruebas que demuestran la gran capacidad de vuelo y potencial dinámico que tienen para realizar tareas complejas y movimientos precisos [3, 4].

En comparación con otras plataformas, el cuadricóptero posee una gran robustez y simplicidad mecánica debido al mínimo número de piezas móviles involucradas [5, 6]. Asimismo se caracteriza por una alta relación empuje-peso que no sólo le permite realizar movimientos ágiles y veloces [7, 8], sino que también le permite transportar grandes cargas [9, 10]. El hecho de que por lo general tenga dimensiones pequeñas, le otorgan la capacidad de maniobrar en espacios reducidos y sortear obstáculos con relativa facilidad [11]. Y por si fuera poco, cuenta con el beneficio de un despegue y aterrizaje vertical, lo que es muy deseable en robots de tipo aéreo [7, 8].

Como es de esperar, el cuadricóptero también posee varios inconvenientes. Desafortunadamente, tiene una capacidad de vuelo muy limitada debido al abrumador consumo de energía de los motores [6, 9]. Pero el problema más relevante radica en la no linealidad y la inestabilidad, características inherentes a su naturaleza que dificultan en gran medida su modelado y posterior control [2, 5, 7, 12].

La operación de controlar el movimiento de estos robots resulta compleja debido a que son sistemas subactuados [2, 8, 12, 13], cualidad que describe a los sistemas mecánicos que no se pueden dirigir para que sigan trayectorias arbitrarias en un espacio de configuración. Por lo general, dicha condición está asociada a los sistemas que poseen un número menor de actuadores que de grados de libertad, y el cuadricóptero, que dispone de seis grados de libertad (tres de traslación y tres de rotación) y cuatro variables de control (una por cada rotor), no es ajeno a ello [14].

1.2 Justificación

Los cuadricópteros son una plataforma robótica muy versátil con infinidad de aplicaciones, especialmente en ambientes peligrosos y áreas de difícil acceso [9]. Entre los usos más relevantes se encuentran la búsqueda y rescate de personas, la detección y control de incendios, el análisis atmosférico para pronósticos meteorológicos, el monitoreo y gestión de cultivos, la inspección de instalaciones eléctricas, la seguridad y vigilancia fronteriza, el transporte y entrega de mercancías, la exploración y mapeo de edificaciones y áreas urbanas, la fotografía y filmación en cine y deportes extremos, o simplemente como elemento recreativo y de entretenimiento [5, 7, 10, 12, 15, 16].

El control de cuadricópteros ha sido abordado en un amplio número de investigaciones en la literatura [15], entre los que destacan principalmente los que usan controladores de tipo PID y en menor medida otros enfoques como el Control por Modos Deslizantes (SMC, por sus siglas en inglés) y el Regulador Cuadrático Lineal (LQR, por sus siglas en inglés) [8, 12]. Sin embargo, el uso de técnicas de inteligencia computacional como parte de su control es notoriamente escaso, siendo allí el área de oportunidad para aportar conocimiento y donde radica la importancia del proyecto.

Un enfoque que ha venido cobrando fuerza recientemente es el control inteligente, el cual comprende una serie de técnicas, tomadas de la inteligencia artificial, con las que se pretenden resolver problemas de control inabordables por los métodos clásicos [17]. Este control tiene una gran aceptación sobre problemas derivados de comportamientos impredecibles de los sistemas a controlar o cuando éstos presentan incertidumbre en los parámetros de su configuración.

En particular, este proyecto se enfoca en desarrollar y validar un sistema para el seguimiento de trayectorias en el espacio de un dron aéreo. La estrategia de control se basa en un controlador difuso, alternable con un controlador PID, que permitirá una posterior comparación de desempeño. Para un mejor sustento de los resultados, las pruebas se establecen no solo de manera simulada, sino también bajo un escenario real con ayuda de un ejemplar modelo AR.Drone 2.0 disponible en el laboratorio. En consecuencia, el sistema debe integrar las herramientas necesarias para su implementación.

Cabe destacar que hasta la fecha no existen proyectos relacionados en la institución por lo que la realización del mismo lo convierte en algo novedoso y pionero en su tipo. Además, una investigación en el tema de los drones aéreos es sumamente interesante y a la vez desafiante, por las diversas áreas del saber que involucra, como la robótica y el control de procesos.

1.3 Objetivos

1.3.1 General

Desarrollar un sistema de control automático para la ejecución de trayectorias en drones aéreos.

1.3.2 Específicos

- Definir el modelo completo del dron aéreo.
- Desarrollar el control automático del dron aéreo usando dos enfoques, uno inteligente y otro convencional.
- Implementar un sistema para la ejecución de trayectorias en un dron aéreo.
- Realizar pruebas de seguimiento de trayectorias bajo diferentes condiciones.
- Validar el desempeño del sistema desarrollado empleando ambos enfoques.

1.4 Antecedentes

En la literatura existe infinidad de trabajos relacionados con el control de los cuadricópteros. Entre los primeros se encuentra [18], en el cual se desarrolló un controlador realimentado dinámico para controlar y desacoplar el movimiento rotacional y el traslacional del cuadricóptero. Uno muy conocido es [19] que utiliza un controlador de seguimiento geométrico directamente sobre un grupo especial euclidiano.

También existen varios trabajos que incluyen control moderno. En [20] proponen un algoritmo de control de estabilización basado en el análisis de Lyapunov. Mientras, en [21] hacen una comparación entre un PID ajustado por el método ITAE (*Integral Time-weighted Absolute Error*), un controlador clásico LQR y un PID ajustado con un lazo LQR. De manera similar, en [12] comparan un control óptimo LQR simple y otro

con ganancia programada (*Gain Scheduling*, en inglés), un control de linealización por realimentación (*Feedback Linearization*, en inglés) y un SMC.

Están otros tipos de controles también. Por ejemplo los que aplican un enfoque *Backstepping* para diseñar el controlador y estabilizar los sistemas de posición y orientación [22]. Tambień, en [8] presentan un enfoque de control usando la técnica de Control Predictivo basado en Modelo (MPC, por sus siglas en inglés) e incorporando restricciones a la entrada y la salida. Asimismo, en [23] proponen una estrategia prealimentada (*Feedforward*, en inglés) con adaptación de los parámetros de movimiento enviados al vehículo. Totalmente distinto es en [24], donde utilizan un control basado en la metodología PVA para obtener acciones de control suaves y aplican un modelo del sistema multi-cuerpos.

Otros trabajos combinan diferentes controles para la traslación y rotación, como [9] donde proponen un MPC para seguir la trayectoria junto a un controlador no lineal H_∞ para estabilizar la rotación. Existe una síntesis muy completa de controladores PID, LQR, H_∞, algoritmos con linealización por realimentación, *Backstepping*, SMC y controles adaptativos, en [2].

Por el lado de los controles inteligentes, los aportes son relativamente recientes y están más vinculados con la lógica difusa. Por ejemplo se acotan [25, 26], que con un controlador basado en lógica difusa, calculan la potencia final de los motores para controlar la altura y los tres ángulos de orientación. Otro caso es [27], donde utilizan un controlador de lógica difusa para mantener la suspensión de un cuadricóptero ante diversas perturbaciones.

En algunos trabajos se pueden ver mezclas y combinaciones entre controladores. Así ocurre con [16] donde crean un controlador híbrido PD-Difuso con conjuntos difusos gaussianos y lo comparan con un controlador PD clásico. También está [10] cuyo diseño de un controlador PID auto-ajustable basado en lógica difusa es capaz de realizar seguimiento de trayectorias cuadradas con variaciones en la carga.

Otras áreas de la inteligencia artificial aplicadas a cuadricópteros no son tan numerosas. En redes neuronales está [28], donde los autores proponen el uso de modelos híbridos de redes neuronales supervisadas para el modelado de sistemas dinámicos complejos de vehículos aéreos. También están las estrategias que requieren

de trayectorias periódicas para su funcionamiento como [4, 5] los cuales emplean un esquema de aprendizaje iterativo que compensa los errores de seguimiento de trayectorias durante la ejecución periódica.

Los trabajos señalados hasta ahora realizaron estudios enteramente simulados, sin embargo, hay investigaciones que llevan el estudio mas allá con la implementación sobre un cuadricóptero real. Para la plataforma 3DR Arducopter, en [2] diseñan un SMC para el seguimiento de trayectorias lineales, circulares y espirales. Mientras que para un cuadrirrotor SO4, en [29] hacen un estudio de dos técnicas de control, un enfoque clásico PID asumiendo dinámicas simplificadas y una técnica moderna LQR basado en un modelo más completo. También en [30], un cuadrirrotor Quanser Qball es utilizado para implementar un control óptimo L_1 basado en el método de factorizaciones estables.

En relación a la plataforma AR.Drone, se pueden nombrar algunos trabajos de investigación que lo involucran. Por ejemplo, [31] es un estudio hecho en MatLab de un control PID en cascada como solución a la estabilización angular y la posición deseada de un AR.Drone 2.0 en trayectorias de tipo circular, linear y oscilante. Otro caso es [32], donde los autores presentan un controlador PD no lineal con polos iguales aplicado sobre un AR.Drone en trayectorias espirales planeadas. Inclusive, en [33] utilizan datos experimentales del AR.Drone para generar un modelo no lineal estático combinado con un modelo dinámico lineal, a partir de los cuales diseñan un controlador de compensación en paralelo.

Específicamente trabajos que involucren el estudio de controladores difusos aplicados al AR.Drone, son muy escasos. Dos ejemplos claros son [34] y [35], ambos implementados sobre el mismo software comercial LabVIEW que emplea sólo las mediciones de los sensores como realimentación. El primero estudia el vuelo desde una posición inicial a una posición deseada, mientras que el otro implementa trayectorias en linea recta, linea recta con un giro perpendicular y curvas.

Así mismo, existen otros trabajos relevantes para la investigación, aquellos sistemas que integren no solo la parte de control, sino también la estimación de la posición y orientación del cuadricóptero. Tal es el caso de [36], donde los autores implementan un estimador de velocidad que usa datos de los sensores inerciales internos y de un láser escáner a bordo para el movimiento horizontal de un MikroKopter. Muy similar ocurre en [37], donde se muestra un sistema de navegación para vehículos

micro-aéreos, que requiere de un buscador de rango láser plano y una unidad de medición inercial para la estimación de estado.

Concretamente, sistemas que hayan empleado la plataforma AR.Drone en la estimación resaltan pocos. Por ejemplo, en [38] emplean un sensor Kinect fijado al techo para estimar la posición horizontal, el sensor ultrasónico del vehículo para estimar la altura, y dos computadoras, una para determinar la localización del vehículo a partir de estos datos, y otra para ejecutar el control de posición. Muy parecido es también [1], que emplea doce cámaras para la localización, y dos computadoras, una se encarga de los cálculos de visión y la otra de los del control. Asimismo, en [6] diseñan un control utilizando la técnica de *Backstepping* para el seguimiento de trayectorias en el plano XY con un AR.Drone 2.0, siendo validado mediante un sistema de comunicación y control junto a un sistema de vision artificial conformado por doce cámaras.

Si se destacan aquellos sistemas que usen los sensores y cámaras del AR.Drone para la estimación y el control, se tienen únicamente los siguientes. Está [39] donde utilizan la cámara frontal y marcas visuales, junto a un filtro de Kalman extendido para estimar la posición global, y un controlador PID para la parte de control. También está [15], donde aplican un controlador no lineal basado en la teoría de Lyapunov, y un filtro de Kalman que fusiona los datos inerciales y visuales para conseguir la posición del vehículo. De último está [40], el cual incorpora un controlador PID para las señales de control y un sistema de navegación visual para la estimación en ambientes desconocidos, compuesto por un filtro de Kalman extendido para la fusión de los datos y una cámara monocular para el SLAM (*Simultaneous Localization And Mapping*) que, a diferencia del resto, no requiere marcas visuales.

El estado del arte demuestra que hay muchos trabajos similares al que aquí se plantea, sin embargo hasta la fecha ninguno reúne las mismas características. El sistema propuesto permite realizar pruebas simuladas y experimentos reales con un AR.Drone 2.0, utilizando únicamente los sensores y cámaras del cuadricóptero para su estimación de estado, e incorporando en el control un controlador difuso, mismo que puede ser alternable con un controlador PID para fines comparativos.

Capítulo 2

Fundamentos Teóricos

En el segundo capítulo se explican las bases teóricas necesarias para abordar el problema del seguimiento de trayectorias sobre drones aéreos. En primera instancia se hace una descripción del cuadricóptero y su modo de funcionamiento. Luego se deduce el modelo matemático completo que rige su comportamiento, se describen las características del AR.Drone 2.0 y se hace referencia a las trayectorias en el espacio. Por último se habla del control automático, del control inteligente y de algunas técnicas de control relevantes para la investigación.

2.1 Cuadricóptero

Según la Real Academia Española (RAE), un dron es una aeronave que vuela sin tripulación [41]. Estos drones, también conocidos como UAVs (*Unmanned Aerial Vehicle*), pueden controlarse desde una ubicación remota o también pueden volar de forma autónoma con ayuda de sistemas complejos de automatización dinámica sobre la base de planes de vuelo preprogramados [42].

Existen varias clasificaciones de los drones aéreos, aunque dependiendo su modo de operación se pueden encontrar dos grandes grupos principalmente: los de tipo avión y los de tipo multicóptero. Los primeros se caracterizan por utilizar alas para su sustentación, y son capaces de desplazarse a grandes velocidades gracias a su diseño

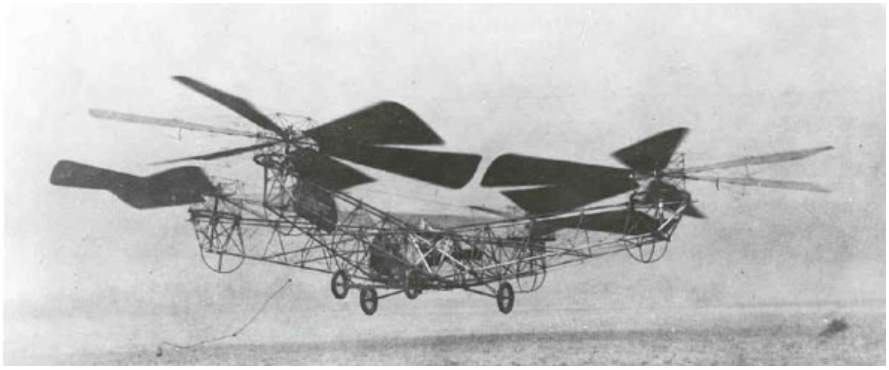


Figura 2.1 Cuadricóptero de Bothezat en 1923 [43]

aerodinámico. Por su parte, los segundos emplean hélices para mantenerse en el aire, y tienen la capacidad de quedar suspendidos en la misma posición del espacio, lo cual es deseable para realizar trayectorias de manera precisa.

Un cuadricóptero o cuadrirrotor es un multicóptero que posee cuatro rotores para su sostén y propulsión [42]. Cada rotor, compuesto por un motor y una hélice, se ubica en la extremidad de un brazo junto a un modulo electrónico de control de velocidad. La configuración en forma de cruz de los brazos constituye la estructura de soporte para el cuerpo del robot. Este último contiene la placa principal, la batería, los sensores y por lo general un transmisor inalámbrico para permitir su control remoto.

El funcionamiento del cuadricóptero es un tanto complejo debido a la mezcla de componentes mecánicos, eléctricos e informáticos. La estabilidad en la sustentación se consigue al equilibrar el empuje de todos los rotores, en otras palabras, que roten a la misma velocidad. Sin embargo, la dirección de rotación entre ellos difiere para cancelar los efectos aerodinámicos y giroscópicos consecuentes. Una pareja de rotores diametralmente opuestos gira en sentido horario mientras que la pareja restante lo hace en sentido antihorario [14, 22].

Ahora bien, se pueden obtener movimientos simples al variar la velocidad relativa de cada rotor ya que se modifica la contribución individual de empuje y torque aplicada al cuerpo [22]. Por lo tanto, si se desea cambiar de altitud, los cuatro rotores deben aumentar o disminuir su velocidad en la misma proporción según sea el caso [8, 29]. En cambio, para inclinar o virar la plataforma se debe aumentar y disminuir en igual medida la velocidad de los rotores que produzcan un torque a favor y en contra respectivamente [7]. La combinación de estas acciones producen movimientos compuestos capaces de

realizar desplazamientos en todo el espacio.

Como dato adicional, el primer cuadricóptero de la historia fue construido en 1907 por los hermanos Louis y Jacques Bréguet, bajo la guía del profesor Charles Richet. Sin embargo, no fue hasta 1922 que se hizo volar un cuadrirrotor poco más de 5 metros del suelo, obra del norteamericano George de Bothezat (Figura 2.1).

2.2 Modelo Matemático

La modelación tiene que ver con la representación de un sistema real, de tal forma que permita explicar o predecir algunos aspectos de su comportamiento. Un modelo matemático se describe en forma de relaciones entre las variables y los parámetros asociados a las propiedades y características dinámicas del sistema.

Siguiendo el orden de ideas, un concepto relevante en el modelado de sistemas relacionado a cuerpos que se mueven por el espacio, es el de grados de libertad o DOF (*Degrees Of Freedom*). Éstos se refieren al numero mínimo de parámetros independientes que se necesitan para definir el estado de un sistema físico. Un cuerpo rígido en tres dimensiones, sin restricciones y aislado del entorno, tiene asociado seis grados de libertad que corresponden a tres movimientos de traslación (ejes X, Y y Z) y tres movimientos de rotación en torno a ellos (ángulos Roll, Pitch y Yaw). En relación a estos últimos, la investigación se restringió únicamente a los ángulos de Euler con la convención ZYX, también conocidos como ángulos de Tait-Bryan. Todo el fundamento matemático que sigue, se sustenta en [42, 44, 45].

La deducción del modelo del cuadricóptero toma en cuenta estas suposiciones:

- La estructura del cuerpo es rígida y simétrica.
- El centro de masa coincide con el origen del marco de referencia del cuerpo.
- Los brazos están alineados con las diagonales del plano XY del marco de referencia del cuerpo.
- Las hélices son idénticas y no sufren deformación al girar.

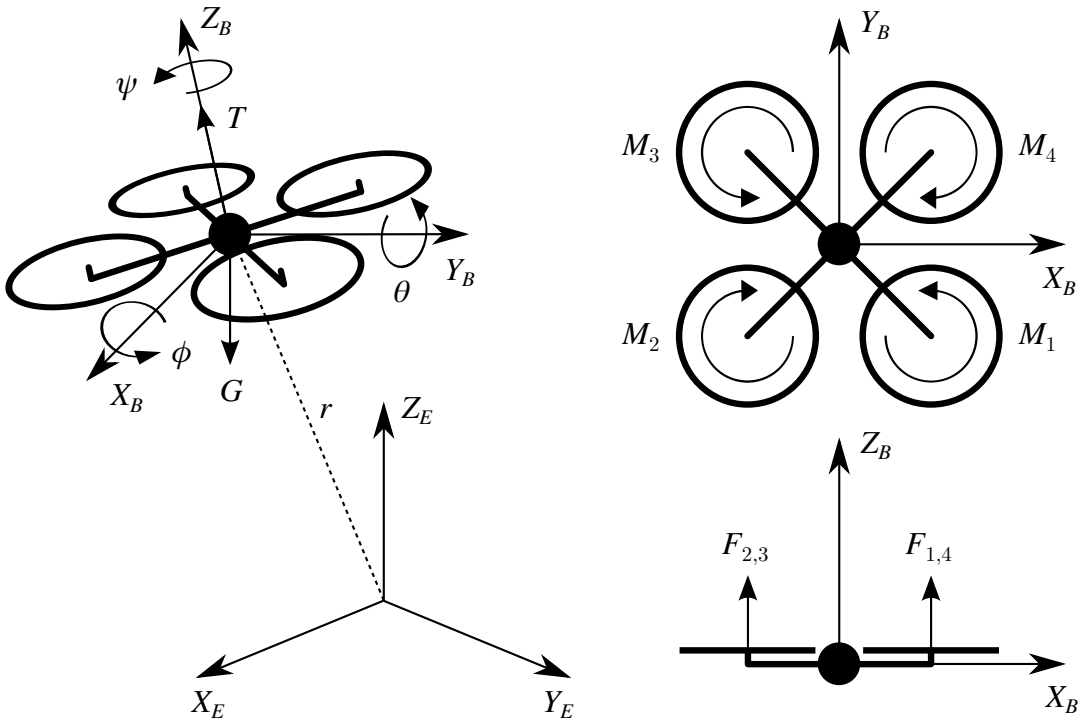


Figura 2.2 Representación del cuadricóptero en el espacio tridimensional, junto a la perspectiva de los planos superior y frontal de su marco de referencia

El estado de un cuadricóptero se define por la posición $r = \begin{bmatrix} x & y & z \end{bmatrix}^T$ y la orientación $\eta = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$ con respecto al marco de referencia global. En la Figura 2.2 se puede observar la ubicación del marco de referencia del cuerpo $<X_B,Y_B,Z_B>$ y la del marco de referencia global $<X_E,Y_E,Z_E>$, así como una representación del cuadricóptero con las fuerzas y momentos que involucra.

Las velocidades lineales y angulares bajo el marco de referencia del cuerpo, están determinadas por $v = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T$ y $\omega = \begin{bmatrix} \omega_\phi & \omega_\theta & \omega_\psi \end{bmatrix}^T$ respectivamente, y es posible relacionarlas al marco de referencia global mediante (2.1) y (2.2), en donde R y W son las matrices de transformación lineal para cada una.

$$\dot{r} = Rv \tag{2.1}$$

$$R = \begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \cos\theta\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

$$\dot{\eta} = W^{-1}\omega \tag{2.2}$$

$$W = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & \sin\phi & \cos\phi\cos\theta \end{bmatrix}$$

La matriz W es invertible sólo si $\theta \neq (2k-1)\,\pi/2$ con $k \in \mathbb{Z}$. Esta limitación se puede sortear si se utiliza en su lugar la ecuación (2.3), en donde $[\omega]_\times$ es la matriz de producto cruzado, y que por medio de la matriz R, relaciona la velocidad angular en el marco de referencia del cuerpo con los ángulos del marco de referencia global.

$$\dot{R} = R\,[\omega]_\times \tag{2.3}$$

$$[\omega]_\times = \begin{bmatrix} 0 & -\omega_\psi & \omega_\theta \\ \omega_\psi & 0 & -\omega_\phi \\ -\omega_\theta & \omega_\phi & 0 \end{bmatrix}$$

Por otro lado, cada rotor i crea una fuerza F_i en dirección al eje del rotor y un momento M_i entorno al mismo eje. Las expresiones (2.4) y (2.5) relacionan ambas variables con la velocidad angular del rotor Ω_i, en donde k_l es el coeficiente de sustentación y k_d es el coeficiente de arrastre de las hélices.

$$F_i = k_l \Omega_i^2 \tag{2.4}$$

$$M_i = k_d \Omega_i^2 \tag{2.5}$$

Las fuerzas combinadas de los rotores crean un empuje en dirección del eje Z del cuerpo T_z, expresado en (2.6), mismo que corresponde al empuje total producido sobre el cuadricóptero $T = \begin{bmatrix} 0 & 0 & T_z \end{bmatrix}^T$ en su marco de referencia. Estas fuerzas de los rotores, generan también los torques en dirección de los ángulos Roll y Pitch del cuerpo τ_ϕ y τ_θ, por medio de (2.7) y (2.8), en donde l es la distancia entre el rotor y el centro de masa del cuadricóptero. En contra parte, el torque en dirección del ángulo Yaw τ_ψ sólo es afectado por los momentos de los rotores, tal como se observa en (2.9). La agrupación de estos tres torques, describen el torque total aplicado sobre el cuadricóptero $\tau = \begin{bmatrix} \tau_\phi & \tau_\theta & \tau_\psi \end{bmatrix}^T$.

$$T_z = F_1 + F_2 + F_3 + F_4 \tag{2.6}$$

$$\tau_\phi = \frac{\sqrt{2}}{2} l \left(-F_1 - F_2 + F_3 + F_4\right) \tag{2.7}$$

$$\tau_\theta = \frac{\sqrt{2}}{2} l \left(-F_1 + F_2 + F_3 - F_4\right) \tag{2.8}$$

$$\tau_\psi = M_1 - M_2 + M_3 - M_4 \tag{2.9}$$

Con relación a las propiedades extrínsecas del cuerpo, la masa está representada por m y el momento de inercia por la matriz

$$I = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}.$$

En este punto existen dos vertientes o enfoques de la mecánica para relacionar el empuje y el torque total de los rotores con la posición y la orientación del cuadricóptero.

Según el enfoque de la mecánica clásica, la dinámica de traslación y rotación de un cuerpo rígido se describe con las ecuaciones de Newton-Euler. Se sabe que en el marco de referencia del cuerpo, la aceleración lineal de la masa es igual a la suma de fuerzas que actúan sobre ella, siendo representadas en este caso, por el empuje de los rotores y la fuerza gravitacional $G = \begin{bmatrix} 0 & 0 & mg \end{bmatrix}^T$ referida a dicho marco de referencia. En consecuencia

$$m\dot{v} = R^{-1}G + T.$$

Transformando la expresión al marco de referencia global con (2.1), se obtiene

$$m\ddot{r} = G + RT. \tag{2.10}$$

Nuevamente en el marco de referencia del cuerpo, se sabe que la aceleración angular de la inercia aunada a los momentos de fuerza centrípetos son equivalentes al torque externo aplicado, es decir,

$$I\dot{\omega} + \omega \times (I\omega) = \tau. \tag{2.11}$$

Algunos autores agregan un elemento adicional Γ del lado izquierdo de la igualdad que representa los momentos de fuerza giroscópicos, el cual está definido por la expresión que sigue, donde I_r es el momento de inercia del rotor y u_z es el vector unitario $\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$.

$$\Gamma = I_r \; \omega \times u_z \left(\Omega_1 - \Omega_2 + \Omega_3 - \Omega_4\right)$$

Obtenida la aceleración angular en el marco de referencia del cuerpo, se convierte al marco de referencia global con ayuda de (2.2), quedando

$$\ddot{\eta} = \frac{d}{dt}\left(W^{-1}\right)\omega + W^{-1}\dot{\omega}. \tag{2.12}$$

$$\frac{d}{dt}\left(W^{-1}\right) = \begin{bmatrix} 0 & \dot{\phi}\cos\phi\tan\theta + \dot{\theta}\sin\phi\sec^2\theta & -\dot{\phi}\sin\phi\cos\theta + \dot{\theta}\cos\phi\sec^2\theta \\ 0 & -\dot{\phi}\sin\phi & -\dot{\phi}\cos\phi \\ 0 & \dot{\phi}\cos\phi\sec\theta + \dot{\theta}\sin\phi\tan\theta\sec\theta & -\dot{\phi}\sin\phi\sec\theta + \dot{\theta}\cos\phi\tan\theta\sec\theta \end{bmatrix}$$

Según el enfoque de la mecánica Lagrangiana, la evolución de un sistema físico se describe mediante las soluciones a las ecuaciones de Euler-Lagrange asociadas al sistema. La ecuación para fuerzas y momentos externos, está dada por (2.13), en donde L es el lagrangiano del sistema y q son las componentes lineales y angulares que intervienen en él, específicamente la posición y la orientación.

$$\begin{bmatrix} F_E \\ M_E \end{bmatrix} = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} \tag{2.13}$$

Este lagrangiano se define como la diferencia entre la energía cinética y la energía potencial del sistema (2.14). Debido a que las componentes lineales y angulares de q no dependen entre si, se pueden estudiar por separado.

$$L\left(q, \dot{q}\right) = E_c\left(q, \dot{q}\right) - E_p\left(q, \dot{q}\right) \tag{2.14}$$

En el caso lineal, las energías cinética y potencial están dadas por

$$E_c = \frac{1}{2}\dot{r}^T m\dot{r} \quad \text{y} \quad E_p = mgz.$$

Se sabe que la fuerza externa está representada por el empuje total de los rotores transformado al marco de referencia global, por lo que al sustituir y operar en (2.13) viene quedando la expresión (2.15) que es equivalente a la (2.10).

$$RT = m\ddot{r} + G \tag{2.15}$$

En el caso angular, sólo existe contribución a la energía cinética ya que la potencial es nula. Para el marco de referencia del cuerpo, la energía cinética se da por

$$E_c = \frac{1}{2}\omega^T I\omega.$$

Dado que el marco de referencia global es el de interés, se deben mover hacia él con ayuda de (2.2) y la matriz Jacobiana J, resultando en

$$E_c = \frac{1}{2}\dot{\eta}^T J\dot{\eta}.$$

$$J = W^T I W$$

Sustituyendo lo anterior en la ecuación (2.13) y, tomando en cuenta que los momentos externos vienen dados por el torque total de los rotores, se deduce la expresión (2.16), donde $C(\eta, \dot{\eta})$ es la matriz de Coriolis que contiene los efectos giroscópicos y centrípetos.

$$\tau = J\ddot{\eta} + C(\eta, \dot{\eta})\,\dot{\eta} \tag{2.16}$$

$$J = \begin{bmatrix} I_x & 0 & -I_x \sin\theta \\ 0 & I_y \cos^2\phi + I_z \sin^2\phi & (I_y - I_z)\sin\phi\cos\phi\cos\theta \\ -I_x \sin\theta & (I_y - I_z)\sin\phi\cos\phi\cos\theta & I_x \sin^2\theta + I_y \sin^2\phi\cos^2\theta + I_z \cos^2\phi\cos^2\theta \end{bmatrix}$$

$$C(\eta, \dot{\eta}) = \dot{J} - \frac{1}{2}\frac{\partial}{\partial\eta}(\dot{\eta}^T J)$$

Los elementos de la matriz C_{ij}, donde i es la fila y j la columna, son:

$$
\begin{aligned}
C_{11} &= 0 \\
C_{12} &= \frac{1}{2}\left(I_y - I_z\right)\dot{\theta}\sin\left(2\phi\right) + \frac{1}{2}\left(I_z - I_y\right)\dot{\psi}\cos\left(2\phi\right)\cos\theta \\
C_{13} &= \frac{1}{2}\left(I_z - I_y\right)\dot{\psi}\sin\left(2\phi\right)\cos^2\theta + \frac{1}{2}\left(I_z - I_y\right)\dot{\theta}\cos\left(2\phi\right)\cos\theta - I_x\dot{\theta}\cos\theta \\
C_{21} &= \frac{1}{2}I_x\dot{\psi}\cos\theta \\
C_{22} &= \left(I_z - I_y\right)\dot{\phi}\sin\left(2\phi\right) + \frac{1}{4}\left(I_y - I_z\right)\dot{\psi}\sin\left(2\phi\right)\sin\theta \\
C_{23} &= \frac{1}{4}\left(I_z - I_y\right)\dot{\theta}\sin\left(2\phi\right)\sin\theta + \frac{1}{2}I_y\dot{\psi}\sin^2\phi\sin\left(2\theta\right) + \frac{1}{2}I_z\dot{\psi}\cos^2\phi\sin\left(2\theta\right) \\
&\quad - \frac{1}{2}I_x\dot{\psi}\sin\left(2\theta\right) + \left(I_y - I_z\right)\dot{\phi}\cos\left(2\phi\right)\cos\theta + \frac{1}{2}I_x\dot{\phi}\cos\theta \\
C_{31} &= -I_x\dot{\theta}\cos\theta \\
C_{32} &= \frac{1}{2}\left(I_z - I_y\right)\dot{\theta}\sin\left(2\phi\right)\sin\theta + \left(I_y - I_z\right)\dot{\phi}\cos\left(2\phi\right)\cos\theta \\
C_{33} &= \left(I_y - I_z\right)\dot{\phi}\sin\left(2\phi\right)\cos^2\theta - I_y\dot{\theta}\sin^2\phi\sin\left(2\theta\right) - I_z\dot{\theta}\cos^2\phi\sin\left(2\theta\right) \\
&\quad + I_x\dot{\theta}\sin\left(2\theta\right)
\end{aligned}
$$

Por último, se debe acotar que las ecuaciones vinculadas a la traslación (2.10) y (2.15) no incluyen el efecto de la resistencia del aire. Con la premisa de representar un sistema lo más realista posible, se incorpora como una fuerza de acción contraria al movimiento, generando así la ecuación (2.17), para la cual k_f es el coeficiente de fricción o resistencia del aire.

$$m\ddot{r} = RT - G - \frac{1}{2}k_f\dot{r}\left|\dot{r}\right| \tag{2.17}$$

2.3 Parrot AR.Drone 2.0

La plataforma experimental elegida para este proyecto es el cuadrirrotor AR.Drone en su versión 2.0 (Figura 2.3), fabricado y comercializado por la empresa francesa Parrot Inc desde 2013 [46]. Originalmente el dispositivo se diseñó como un juguete de alta tecnología, pero que pronto encontró utilidad en otros campos debido a sus excelentes prestaciones [15].

Figura 2.3 Parrot AR.Drone 2.0

Existe una aplicación oficial llamada AR.Freeflight, disponible gratuitamente para dispositivos móviles iOS y Android, que permite controlar el AR.Drone 2.0 vía conexión Wifi b/g/n. Parrot también proporciona un conjunto de herramientas de software o SDK, que facilita a los usuarios el desarrollo de aplicaciones para la comunicación y control de la plataforma [47, 48].

La estructura exterior del AR.Drone 2.0 se compone por tubos de fibra de carbono, piezas de plástico nylon con 30 % de fibra de vidrio, nanorevestimiento repelente a líquidos y espuma para aislar el centro inercial de las vibraciones producidas por los motores [49]. Posee también dos cascos de protección hechos de polipropileno expandido, uno para exteriores de 45×45 cm y otro para interiores de 52×52 cm, que le confieren un peso total de 380 y 420 g respectivamente [50].

En especificaciones técnicas, tiene un procesador ARM Cortex A8 de 32 bits a 1 GHz de frecuencia, una memoria RAM tipo DDR2 de 1 GB a 200 MHz y un sistema operativo Linux embebido versión 2.6.32 [51]. Asimismo incorpora dos cámaras de vídeo, una HD 720p de 30 FPS con 92° de diagonal situada en la zona frontal y otra QVGA de 60 FPS con 64° de rango ubicada en la parte inferior [49]. Emplea como fuente de energía una batería de 11.1 V y 1000 $\text{mA} \cdot \text{h}$ que le otorga una autonomía aproximada de 12 min de vuelo [51]. De manera auxiliar posee un puerto USB 2.0 que permite grabar vídeo en una memoria USB o también conectar un GPS para una localización global [50].

Como actuadores usa 4 motores tipo inrunner sin escobillas de 14.5 W de potencia y 28500 rpm, cada uno con controladores AVR de 8 MIPS [52]. En cuestión de sensores, este modelo está equipado con un giroscopio de 3 ejes con precisión de 2000°/s, un acelerómetro de 3 ejes con precisión de 50 mg, un magnetómetro de 3 ejes con precisión de 6°, y dos restantes, un barómetro con una precisión de 10 Pa y un sensor de ultrasonido con un rango efectivo de 6 m, destinados a medir la posición vertical [49]. También se usa la cámara inferior para ayudar a estimar la velocidad lineal del cuerpo [53, 49].

La placa principal es la que gestiona los datos procedentes de los sensores, las transmisiones de vídeo de ambas cámaras, las instrucciones resultantes para los motores y la red de comunicación inalámbrica del dispositivo. El firmware instalado es capaz de realizar automáticamente los procedimientos de despegue, aterrizaje y estabilización de vuelo, además de responder a comandos de movimiento externos [15]. Toda información o instrucción que se envía en la red Wifi, está codificada bajo un protocolo específico que se define a continuación.

Respecto a las señales de control, el protocolo de comunicación establece un vector cuyos elementos son la velocidad lineal en el eje z ($u_{\dot{z}}$), la velocidad angular en dirección Yaw ($u_{\dot{\psi}}$) y las inclinaciones en dirección Roll (u_{ϕ}) y Pitch (u_{θ}), cada uno normalizado en relación al valor máximo configurado [54]. En contra parte, el AR.Drone entrega un vector compuesto por señales de salida, entre las que destacan los ángulos Roll (ϕ), Pitch (θ) y Yaw (ψ) referidos al sistema de coordenadas global, la altura (z) relativa a la superficie debajo de él, y las velocidades lineales en X (v_x) y Y (v_y) con referencia al sistema de coordenadas del cuerpo [54].

2.4 Trayectoria en el espacio

En la mayoría de aplicaciones con cuadricópteros, se requiere planificar trayectorias de vuelo para obtener el movimiento final deseado [13]. Toda trayectoria se define como un conjunto de puntos que representan las coordenadas de posición de una ruta particular en cada instante de tiempo. Por ende, su expresión matemática tiene una función asociada a cada coordenada del espacio y dependiente de la variable tiempo (2.18).

$$f(t) = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} \tag{2.18}$$

A modo de ejemplo, en el Apéndice se lista una recopilación de diferentes trayectorias en el espacio, donde se incluye tanto las funciones generadoras como una representación gráfica de las mismas al final de la sección.

2.5 Control Automático

El objetivo fundamental de un sistema de control consiste en mantener una o más variables de un proceso en un valor determinado o dentro de ciertos límites. Se entiende como proceso al conjunto de elementos que producen un resultado final, y como variable a cualquier magnitud física que pueda sufrir cambios. Si no existe intervención humana para realizar dicha operación, se habla entonces de un control automático [55].

Casi todos los sistemas de control automáticos dependen de una vía de realimentación para comparar el estado actual de la variable con el ideal. Una variable puede apartarse del valor deseado por cambios en las condiciones de trabajo o por efecto de las perturbaciones, que son señales no deseadas capaces de adoptar valores al azar dentro de ciertos límites. Cuando ocurre cualquiera de estos casos, se debe actuar sobre el elemento o elementos del proceso que generan la variable de interés, a fin de minimizar cualquier discrepancia entre ésta y el valor deseado. El componente del sistema de control automático, encargado de sintetizar dichas acciones correctivas, es el controlador [56].

Un esquema de control básico se compone por un controlador y un proceso a controlar, conectados en una estructura de lazo cerrado (Figura 2.4). La señal de salida del proceso corresponde a la variable que se desea controlar, mientras que la señal de referencia es el valor deseado o *set-point* de dicha variable. La diferencia entre ambas produce una señal de error que es utilizada por el controlador para generar la señal de control sobre la entrada del proceso.

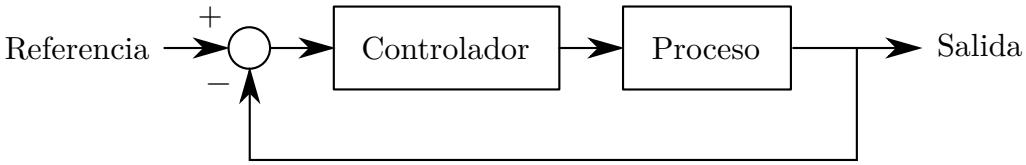


Figura 2.4 Lazo de control básico

El primer trabajo significativo en control automático fue hecho por James Watt en 1774, un regulador centrífugo para el control de la velocidad de una máquina de vapor. Durante los siguientes años, se fueron desarrollando lentamente algunas técnicas de control y análisis de estabilidad, hasta la década de 1950, que numerosas reuniones y publicaciones de libros y artículos consolidaron la teoría del control [57]. Esto ultimo permitió crear una mayor conciencia del potencial del control automático y atrajo la atención de una comunidad técnica mucho más amplia.

El beneficio del control automático se sustenta principalmente en la reducción de costos asociados a la generación de bienes y servicios y en el incremento de la calidad y volúmenes de producción de las industrias. Asimismo, contribuye en la eliminación de errores y el aumento en la seguridad de los procesos. Sin embargo, su aplicación resulta un tanto difícil porque no hay una estrategia de solución garantizada para cada caso [58]. En especial, para procesos complejos, de orden alto, con tiempo muerto y/o corrimiento de fase, es muy importante escoger adecuadamente el controlador y sus parámetros. De no ser así, puede acarrear controles limitados e ineficaces, que producen lentos y cíclicos periodos para estabilizar el sistema, operaciones erróneas y, en el peor caso, un colapso del sistema [55].

Existen características deseables en los sistemas de control para que presten alguna utilidad. La más importante y elemental es la estabilidad, capacidad de un sistema de generar una respuesta de magnitud acotada ante una entrada o perturbación también acotada, muy útil para establecer si su comportamiento tenderá a un estado de equilibrio. También está la robustez, capacidad de tolerancia o insensibilidad ante los cambios o perturbaciones que puedan presentarse, la exactitud, capacidad de mantener las variables de interés dentro de un entorno tolerable o con un error pequeño, y la velocidad de respuesta, capacidad de adecuación dinámica del sistema a condiciones cambiantes [56].

En la literatura se han desarrollado infinidad de enfoques de control automático. Algunos de los más destacados son el Realimentado (*Feedback*, en inglés), el Prealimentado (*Feedforward*, en inglés), el *Backstepping*, el PID (*Proportional Integral Derivative*), el LQR (*Linear Quadratic Regulator*), el SMC (*Sliding Mode Control*), el H_∞ No Lineal, el Robusto, el Óptimo, el Adaptativo, entre muchos otros [55, 59, 60]. Un dato importante es que los controles no son excluyentes entre si, lo que permite la combinación y generación de nuevas y mejores variantes.

2.6 Control Inteligente

Desde el punto de vista teórico, un sistema de control involucra inteligencia si integra realimentación y conocimiento en el proceso de planificación y generación de acciones que conlleven alcanzar los objetivos de control [26]. Bajo esta premisa, el término inteligente está estrechamente ligado al de autónomo, ya que un sistema inteligente por lo general implica tener un alto grado de autonomía [17].

Esencialmente, el control inteligente se enfoca en abordar el problema de control con técnicas de la inteligencia artificial, las cuales tratan de emular las funcionalidades inteligentes de los seres vivos y, en concreto, el proceso del razonamiento humano [17, 26]. La idea nace por la necesidad de encontrar e implementar mejores y más sofisticadas soluciones en el área del control, especialmente en aquellos sistemas complejos donde ciertos requerimientos no pueden ser alcanzados con el control convencional [61].

Los inicios del control inteligente se remontan a 1971, con los trabajos de K. S. Fu [62]. Sin embargo, no alcanzó un importante desarrollo hasta la década de 1990 cuando surgieron numerosos artículos, conferencias y libros relacionados [61, 63, 64]. Desde entonces, la aplicación del control inteligente ha ido creciendo conforme ha avanzado la tecnología, desde el microprocesador hasta el procesamiento en paralelo, que han permitido ampliar y diversificar sus posibilidades [65].

El control inteligente ofrece perspectivas interesantes ya que define metodologías que permiten realizar de forma automática algunas de las tareas realizadas típicamente por los humanos [26]. También, ha probado ser un enfoque efectivo para resolver problemas que involucren incertidumbre, complejidad matemática y gran cantidad de

información [58]. No obstante, muchas veces su uso se ve limitado por los abundantes recursos computacionales que requiere.

Entre las técnicas asociadas más importantes se encuentran: los sistemas expertos, capaces de obtener experiencia a partir de una base de conocimiento, la lógica difusa, capaz de manejar la incertidumbre, imprecisión e inexactitud de los sistemas reales; las redes neuronales, capaces de aproximar el comportamiento de un sistema a partir de un conjunto de ejemplos; y los algoritmos evolutivos, capaces de optimizar parámetros en los sistemas usando analogías bioinspiradas [7, 61, 65]. Algunas veces también, el control inteligente incluye metodologías desarrolladas para el control convencional [17, 26].

2.7 Controlador PID

El controlador PID, desde sus inicios, ha demostrado ser un controlador sobresaliente en una amplia gama de aplicaciones [7]. Al día de hoy, figura como uno de los controladores que goza con mayor popularidad en la industria, en parte debido a que resulta más que suficiente para cumplir los requisitos de control deseados [58]. De hecho, tiene una muy buena robustez aunado a un diseño sencillo con parámetros fáciles de ajustar [7]. No obstante, su uso no es garantía de un control óptimo ni la existencia de estabilidad en el sistema.

El primer registro que se tiene del controlador, es en el artículo *Directional stability of automatically steered bodies* hecho por el científico Nicolás Minorsky en 1922 [66], en donde describe el uso de controladores de tres términos para el control automático de buques. Sin embargo, no fue hasta finales de la década de 1930 que su materialización se hizo realidad, con la introducción del primer controlador de propósito general tipo PID con una acción de control ajustable, obra de la *Taylor Instrument Company.*

La Figura 2.5 muestra la estructura de un controlador PID. Éste se caracteriza por presentar tres componentes de acción importantes, los cuales se describen en detalle a continuación [67]:

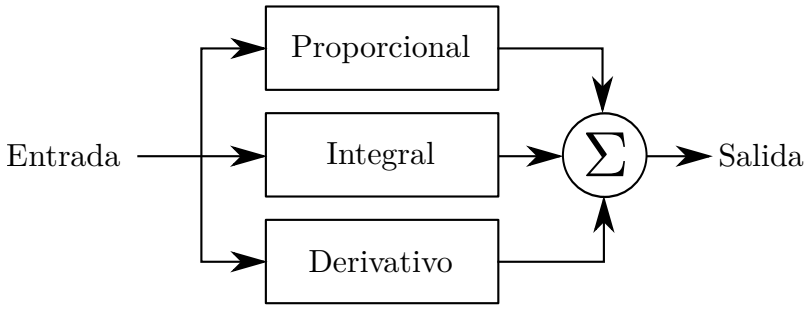


Figura 2.5 Esquema del controlador PID

- La acción proporcional produce una salida que es función directa del valor actual de error, lo que la vincula con la sensibilidad que posee el controlador. Si su contribución es muy alta, el sistema puede llegar a ser inestable, ya que un pequeño aumento del error resultaría en una respuesta de salida desmesurada. En contra parte, si es muy baja, la acción de control puede ser insuficiente para compensar las perturbaciones del sistema.

- La acción integral tiene relación con la suma del error instantáneo en el tiempo, lo que permite lograr un mejor seguimiento de la duración del error. Su inclusión acelera el movimiento del proceso hacia el punto deseado y elimina el error de estado estacionario residual. Sin embargo, dado que su contribución responde a los errores acumulados en el pasado, puede provocar que el valor actual sobrepase el valor de referencia, aparte que tiene un efecto desestabilizador por el corrimiento de fase agregado.

- La acción derivativa otorga propiedades predictivas sobre el comportamiento del sistema debido a su nexo con la pendiente del error en el tiempo, lo que ayuda a prevenir sobrepicos y a mejorar el tiempo de estabilización del sistema. No obstante, debe utilizarse cuidadosamente para evitar que amplifique el ruido de medición o del proceso, el cual puede producir grandes variaciones aleatorias en la salida. Además, no se recomienda incluirla cuando existan grandes retrasos en el bucle ya que genera un efecto negativo en la estabilización.

La suma ponderada se utiliza para ajustar la salida del controlador y proveer de este modo una acción de control diseñada para los requerimientos específicos del sistema.

2.8 Controlador Difuso

Un controlador difuso es aquel que hace uso de la lógica difusa para lograr el objetivo del control. A grandes rasgos, la lógica difusa se refiere a un conjunto de teorías matemáticas que emplean grados de verdad parciales y continuos, en vez de valores de verdad binarios como es en la lógica clásica [10]. Fue propuesta en 1965 por el profesor Lotfi Zadeh [68] como una alternativa para mejorar la capacidad de razonamiento de las máquinas imitando el proceso de razonamiento impreciso o aproximado de los humanos. La idea de asociarla a un controlador no se exploró hasta 1975, con la publicación de los científicos Ebrahim Mamdani y Sedrak Assilian [69].

En su favor, el controlador difuso cuenta con la capacidad de integrar conocimiento humano, además de tener la habilidad de manejar entradas con algún grado de incertidumbre. Sin embargo, para operar correctamente requiere de un buen conocimiento de control en forma de reglas y un apropiado ajuste de sus numerosos parámetros. Asimismo, resulta complejo establecer la estabilidad de los sistemas que incorporen este controlador [58].

Respecto al modo de funcionamiento, un controlador difuso presenta cuatro componentes principales, mismos que se muestran en la Figura 2.6, y se explican como sigue [70]:

- La difusificación que es la etapa que traduce los valores numéricos de entrada al dominio difuso. Cada entrada tiene definida un grupo de conjuntos difusos, los cuales se asocian a un valor de membresía. El valor de membresía resultante dependerá de los conjuntos difusos y del valor de entrada.
- La base o conjunto de reglas que expresan el conocimiento del que se dispone para el control. Están expresadas en formato de proposiciones lógicas condicionales: SI <antecedente>, ENTONCES <consecuente>.
- La máquina de inferencia difusa que es la etapa que determina las salidas difusas a partir de las entradas difusas. Este proceso requiere el uso de operadores lógicos difusos para combinar las reglas predefinidas y así mapear los conjuntos difusos de entrada y salida.

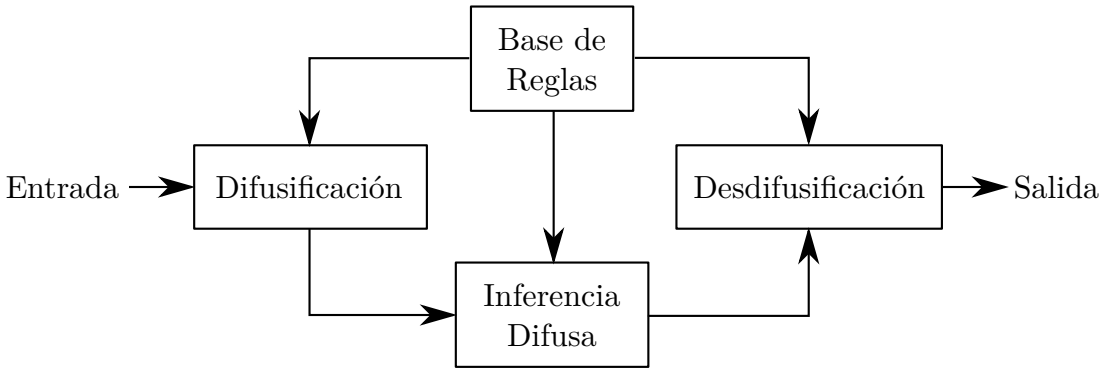


Figura 2.6 Esquema del controlador difuso

- La desdifusificación que es la etapa que traduce la salida difusa en un valor numérico. La salida tiene asociada un grupo de conjuntos difusos, los cuales permitirán calcular un valor numérico a partir del conjunto de valores de membresía difusos.

Comúnmente la salida se obtiene como la combinación ponderada de los valores de membresía resultantes. En cuanto a la estructura y los parámetros del sistema, están asociados al conocimiento que se disponga.

Capítulo 3

Metodología

En el tercer capítulo se definen los aspectos relacionados a la metodología de la investigación. Un sistema de control se desarrolla para dar solución al problema de interés, el cual se descompone en una serie de módulos que se explican y analizan por separado. Al final se detallan los criterios para medir el rendimiento del sistema bajo las diferentes modalidades de estudio.

3.1 Sistema General

La estrategia planteada se enfocó en la creación de un sistema de control que permitiera el estudio de la ejecución de trayectorias por parte de un cuadricóptero. En consecuencia, el sistema debía permitir la integración del control inteligente en que se basa el estudio, y del control convencional que serviría de comparación. Asimismo, su diseño debía ser capaz de facilitar una implementación física posterior sobre el modelo AR.Drone 2.0 presente en el laboratorio.

El desarrollo del sistema fue hecho totalmente en lenguaje C++, debido a que es un lenguaje estándar muy conocido en el desarrollo de aplicaciones y con una extensa documentación al respecto. Además, forma parte de los pilares fundamentales de la programación lo que le confiere una inigualable compatibilidad con diferentes entornos y dispositivos, especialmente en el área de robótica.

Todo aquel que esté familiarizado con el lenguaje, coincidirá que es una herramienta muy potente que requiere una gran cantidad de tiempo y esfuerzo para realizar las más sencillas aplicaciones. De ahí parte la necesidad de utilizar, en la medida de lo posible, librerías estándares (STL) que faciliten y aceleren el proceso de programación. Sin embargo, existen aplicaciones muy específicas donde estas librerías se ven limitadas o no son suficientes, por lo que se requiere elaborar librerías propias o incorporar librerías externas que solventen dichos requerimientos.

Tal es el caso de GLM, una librería matemática en C++ que facilitó el manejo y las operaciones de elementos como vectores y matrices, además de integrar algunas funciones matemáticas complementarias [71]. Otras librerías externas a resaltar son FuzzyLogic, OpenGL y las relacionadas a ROS, de las cuales se habla más adelante en el capítulo.

El código fuente que permite la creación de la aplicación, aplica el estándar C+11 durante el proceso de compilación, por lo que un requerimiento básico en la computadora es que sea capaz de aplicarlo. Un punto a favor es que el proceso no está limitado a un sistema operativo en específico, ya que se realiza por medio de un archivo de configuración Makefile que contiene las instrucciones necesarias para generar el ejecutable. Como dato adicional, para sistemas Unix como Linux y Mac OS se usa el compilador G++ mientras que para sistemas Windows se emplea el compilador Microsoft Visual C++.

Con relación al sistema principal, se establecieron dos modalidades a destacar: el modo simulación, que sirve para realizar el estudio con un modelo simulado del dron y bajo diferentes condiciones simuladas, y el modo implementación, que permite efectuar las pruebas directamente sobre el AR.Drone real. La elección de la modalidad se define al momento de la compilación.

Desde un principio, se optó por dividir el sistema en diferentes módulos a fin de facilitar un correcto desarrollo y una mejor definición de sus funciones. La estructura general se compone por un total de ocho módulos, algunos esenciales y otros prescindibles según su propósito en el sistema. En las Figuras 3.1 y 3.2, se observa por medio de una linea continua o punteada, cuáles módulos son necesarios y cuáles opcionales para las distintas modalidades. Cada módulo se explica en detalle a lo largo de las secciones siguientes.

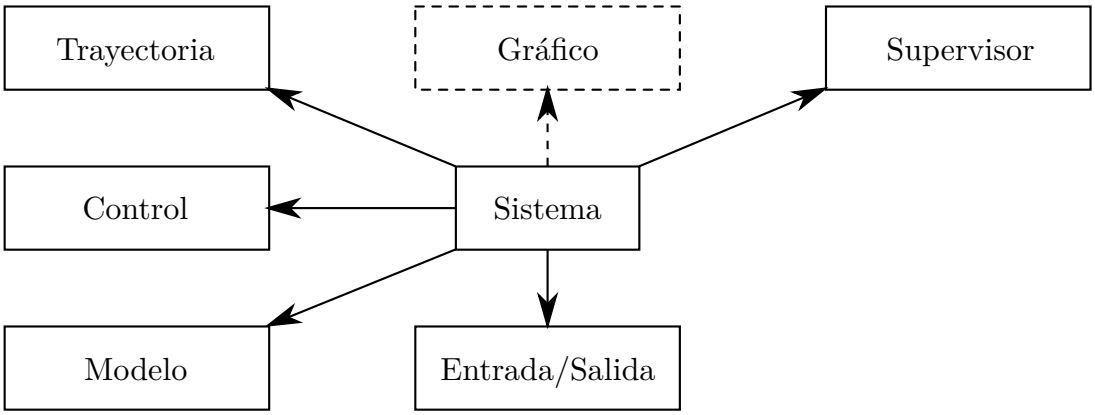


Figura 3.1 Módulos necesarios y opcionales en la simulación

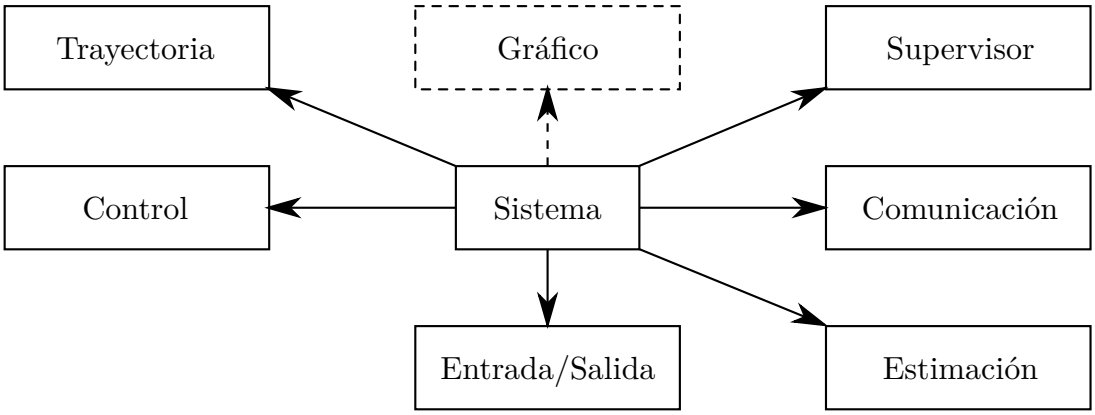


Figura 3.2 Módulos necesarios y opcionales en la implementación

3.2 Módulo de Trayectorias

Un elemento clave para ejecutar trayectorias sobre cuadricópteros es un generador de las mismas. Éste tiene la tarea de definir las rutas de vuelo que logren el objetivo propuesto. También puede incluir aspectos adicionales como que respete la dinámica del vehículo, que esté libre de colisiones y que realice movimientos en función de la detección de ciertos objetos. Para el caso que atañe, sólo se necesita de un generador de trayectorias previamente definidas.

Para cumplir el objetivo, se le suministró al sistema de un conjunto de trayectorias en el espacio que pudieran ser útiles para la investigación. La lista completa se encuentra en el Apéndice del trabajo. A fin de establecer los mismos estándares y

que fueran equiparables, todas las trayectorias se encuentran normalizadas tanto en la entrada ($t \in [0,1]$) como en la salida ($x, y, z \in [-1,1]$). Esta trayectoria primitiva, se pasa después por un proceso de transformación para que se ajuste a los requerimientos de dimensiones y lapso de tiempo deseados para el estudio.

La transformación de la trayectoria se realiza por medio de la expresión (3.1), y se obtiene $r^*(t) = \begin{bmatrix} x^*(t) & y^*(t) & z^*(t) \end{bmatrix}^T$. Allí se observa que previamente la entrada t, que corresponde al tiempo actual, se ajusta a un rango adecuado para la función generadora de la trayectoria primitiva $f(t)$. Esta conversión se realiza con los valores de tiempo inicial t_o y tiempo final t_f de la prueba, lo que los liga a la rapidez de avance de la trayectoria. El resultado obtenido de la función, se le aplican varias operaciones que corresponden a un escalamiento, una rotación y una traslación en ese orden.

$$r^*(t) = c + B\left(Af\left(\frac{t - t_o}{t_f - t_o}\right)\right) \tag{3.1}$$

$$B = \begin{bmatrix} \cos\alpha + u_x^2(1-\cos\alpha) & u_x u_y(1-\cos\alpha) - u_z\sin\alpha & u_x u_z(1-\cos\alpha) + u_y\sin\alpha \\ u_y u_x(1-\cos\alpha) + u_z\sin\alpha & \cos\alpha + u_y^2(1-\cos\alpha) & u_y u_z(1-\cos\alpha) - u_x\sin\alpha \\ u_z u_x(1-\cos\alpha) - u_y\sin\alpha & u_z u_y(1-\cos\alpha) + u_x\sin\alpha & \cos\alpha + u_z^2(1-\cos\alpha) \end{bmatrix}$$

$$A = \begin{bmatrix} a_x & 0 & 0 \\ 0 & a_y & 0 \\ 0 & 0 & a_z \end{bmatrix} \qquad c = \begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix}$$

La primera operación modifica las dimensiones de la trayectoria en dirección a su tamaño final, mediante la matriz diagonal A donde sus elementos escalan cada eje. La segunda operación permite orientar la trayectoria en cualquier dirección, y se logra con la matriz B, generada a partir de un vector unitario de rotación $u = \begin{bmatrix} u_x & u_y & u_z \end{bmatrix}^T$, un ángulo α y la ecuación (3.2). Y la última operación se encarga de mover la trayectoria a su posición final dentro del espacio, siendo dicho lugar el punto c.

$$B = (\cos\alpha)\, I_3 + (\sin\alpha)\, [u]_\times + (1-\cos\alpha)\, uu^T \tag{3.2}$$

Para la ecuación anterior, I_3 es la matriz identidad y $[u]_\times$ es la matriz de producto cruzado, ya explicado en (2.3).

Con todo lo explicado, el generador de trayectoria es capaz de proveer los puntos $x^*(t)$, $y^*(t)$ y $z^*(t)$ que servirán de posición deseada para el control del cuadricóptero. Sin embargo, este último cuenta además con otros tres grados de libertad que requieren ser establecidos. Como se verá más adelante en la sección de control, los valores de los ángulos $\phi^*(t)$ y $\theta^*(t)$ son obtenidos mediante el mismo control, mientras que el ángulo $\psi^*(t)$ sí debe fijarse con antelación. Es posible orientar el frente del cuadricóptero hacia una dirección especifica mientras se realiza la trayectoria, como también hacer que varíe con el movimiento, tal cual como un avión. Dado que no es un objetivo de la investigación, por simplicidad se dejó en un valor fijo nulo.

$$\psi^*(t) = 0 \tag{3.3}$$

3.3 Módulo del Modelo

Antes de efectuar el análisis y posterior control de un sistema dinámico es necesario tener un modelo del proceso. En el capítulo anterior se dedujo el modelo del cuadricóptero, de donde se obtuvieron una serie de alternativas. Principalmente el comportamiento se define por dos pares de Ecuaciones Diferenciales Ordinarias (ODEs), un par para el modelo cinemático que pueden ser (2.1)-(2.2) ó (2.1)-(2.3), y otro par para el modelo dinámico representados por (2.11)-(2.17) ó (2.16)-(2.17). Existen también seis ecuaciones no diferenciales importantes, (2.4) y (2.5) que relacionan la velocidad de los rotores con la fuerza y el momento producido por los mismos, y (2.6), (2.7), (2.8) y (2.9), que vinculan estos últimos con el empuje y el torque aplicado sobre el cuadricóptero.

Cada una de estas ecuaciones fueron incorporadas en la aplicación, pero las que eran redundantes, se inhabilitaron durante el proceso de compilación. Las ecuaciones diferenciales seleccionadas para el estudio fueron (2.1)-(2.2) y (2.16)-(2.17) porque simplemente son más aplicadas en trabajos de investigación relacionados [9, 22, 32, 72, 1, 20, 2]. Respecto a las ecuaciones no diferenciales, todas eran necesarias, por lo que se integraron para generar las siguientes ecuaciones.

$$
\begin{aligned}
T_z &= k_l \left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \\
\tau_\phi &= \frac{\sqrt{2}}{2} l k_l \left(-\Omega_1^2 - \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right) \\
\tau_\theta &= \frac{\sqrt{2}}{2} l k_l \left(-\Omega_1^2 + \Omega_2^2 + \Omega_3^2 - \Omega_4^2\right) \\
\tau_\psi &= k_d \left(\Omega_1^2 - \Omega_2^2 + \Omega_3^2 - \Omega_4^2\right)
\end{aligned}
\tag{3.4}
$$

De aquí se observa que la variable de entrada al modelo son las velocidades de los cuatro rotores, las cuales se transforman en esfuerzo y torque por sustitución directa. Con estos valores y las ecuaciones diferenciales se debe obtener la posición y orientación del cuadricóptero. Sin embargo, esto no se resuelve por sustitución como en el caso anterior ya que hay una dependencia de la repuesta con las variables internas de la expresión. Su solución requiere de otro enfoque.

Para resolver ecuaciones diferenciales ordinarias existen dos grupos de métodos, los explícitos y los implícitos. Los primeros calculan el estado de un sistema al instante siguiente a partir del estado en el instante actual. Por su lado, los segundos encuentran la solución al resolver una ecuación que involucra tanto el estado actual del sistema como el estado siguiente. En comparación, los métodos explícitos son más eficientes porque hacen una aproximación, pero los métodos implícitos son más exactos porque no la hacen. Según la aplicación es mejor escoger un tipo u otro.

En el área de robótica es preferible decantarse por la eficiencia en vez de la exactitud, ya que para aplicaciones en tiempo real se requieren realizar cálculos lo más rápido posible. Estableciendo un tamaño de paso del tiempo Δt lo bastante pequeño, el error de aproximación es despreciable por lo que es perfectamente admisible emplear un método explícito.

Existen varios métodos para cumplir la tarea, pero entre los más conocidos están el de Euler, el de Heun, el del Punto-Medio, el de Runge-Kutta de 4 orden y el de Dormand-Prince. Según el orden mencionado, van aumentando en exactitud pero también implican más cálculos. Se decidió utilizar el método de Runge-Kutta por ofrecer mejor equilibrio entre eficiencia y exactitud. Aun así, los cinco métodos se incluyeron en una librería propia del código, y se puede cambiar la elección al momento de compilar el código.

Por lo tanto, para obtener el estado siguiente $q\left(t+\Delta t\right)$, se emplea una función H que incluye la ecuación diferencial, y requiere como entrada el estado actual $q\left(t\right)$, el tiempo actual t y el tamaño de paso Δt. Para el caso de la posición y la orientación, las ecuaciones están dadas por (3.5).

$$\begin{aligned} r\left(t+\Delta t\right) &= H\left(r\left(t\right), t, \Delta t\right) \\ \eta\left(t+\Delta t\right) &= H\left(\eta\left(t\right), t, \Delta t\right) \end{aligned} \tag{3.5}$$

En otro orden de ideas, se decidió establecer diferentes tipos de perturbaciones aplicadas al modelo para generar casos de estudio interesantes. El primer escenario es el ideal o sin perturbaciones que corresponde tal cual se tiene el modelo hasta ahora. El segundo escenario incluye una perturbación constante externa que afecta al cuadricóptero, representada por la fuerza del viento desplazándose a cierta velocidad. La expresión (3.6) señala el término que se ajustó en (2.17), para incorporar la velocidad del viento w. Y el tercer escenario introduce una perturbación variable, como es el caso del ruido asociado a los sensores que obtienen la posición y orientación. Para ello se usan las ecuaciones (3.7), en donde N es una función generadora de valores aleatorios con distribución normal de media μ y varianza σ^2.

$$\frac{1}{2}k_f\left(w-\dot{r}\right)\left|w-\dot{r}\right| \tag{3.6}$$

$$\begin{aligned} r &= r + N\left(\mu_r, \sigma_r^2\right) \\ \eta &= \eta + N\left(\mu_\eta, \sigma_\eta^2\right) \end{aligned} \tag{3.7}$$

Una vez establecido todo el modelo, no queda mas que especificar los valores de sus parámetros. La Tabla 3.1 muestra el valor de aquellos parámetros que son fijos durante las pruebas, teniendo en cuenta que los mínimos que faltan son iguales a los máximos pero negativos. La información se obtuvo de diferentes fuentes [10, 31, 38, 49, 50, 51, 54, 73], a excepción de los que lógicamente no tienen límite o no hay referencias al respecto. En relación a las velocidad máxima y mínima de los rotores, las cuales corresponden realmente a la velocidad de rotación de las hélices, fue necesario obtener el rango de velocidad de los motores (10350 a 41400rpm) y por medio de la relación de engranajes (1:8.75) llevarlo a la de las hélices.

Tabla 3.1 Parámetros del modelo

Parámetro	Valor	Unidad	Parámetro	Valor	Unidad
g	9.81	$\mathrm{m/s^2}$	m	0.429	kg
I_x	2.238×10^{-3}	$\mathrm{kg \cdot m^2}$	l	0.1785	m
I_y	2.985×10^{-3}	$\mathrm{kg \cdot m^2}$	k_l	8.048×10^{-6}	
I_z	4.804×10^{-3}	$\mathrm{kg \cdot m^2}$	k_d	2.423×10^{-7}	
I_r	2.030×10^{-5}	$\mathrm{kg \cdot m^2}$	k_f	0.15	kg/m
x_{max}	5	m	ϕ_{max}	30	°
y_{max}	5	m	θ_{max}	30	°
z_{max}	5	m	ψ_{max}	∞	°
$\dot{x}_{max}$	5	m/s	$\dot{\phi}_{max}$	∞	°/s
$\dot{y}_{max}$	5	m/s	$\dot{\theta}_{max}$	∞	°/s
$\dot{z}_{max}$	2	m/s	$\dot{\psi}_{max}$	350	°/s
Ω_{max}	495.5	rad/s	Ω_{min}	123.9	rad/s

3.4 Módulo de Control

El control de procesos se ocupa para determinar los valores de entrada al modelo que aseguren un seguimiento lo más fielmente posible de la trayectoria. Para lograrlo, se requiere una estructura de dos lazos de control realimentados en configuración de cascada, donde el control interno se encargue de ajustar la orientación y el externo del seguimiento de la trayectoria. Con base en las ecuaciones que rigen su comportamiento, las entradas de los controladores son el error en las variables de posición y orientación mientras que su salida es la aceleración lineal y angular respectivamente.

Estas aceleraciones deben pasarse inmediatamente por un saturador que restrinja su valor, con lo cual impedir que se produzca una velocidad fuera de los límites y que la posición u orientación resultante se encuentre dentro de los valores permitidos, todo por motivos de seguridad. La expresión que lo hace posible es (3.8), en donde q representa cualquiera de las variables de posición u orientación.

$$\frac{1}{\Delta t}\left(\text{máx}\left\{\dot{q}_{min}, \frac{1}{\Delta t}\left(q_{min}-q\right)\right\}-\dot{q}\right) \leq \ddot{q} \leq \frac{1}{\Delta t}\left(\text{mín}\left\{\dot{q}_{max}, \frac{1}{\Delta t}\left(q_{max}-q\right)\right\}-\dot{q}\right) \tag{3.8}$$

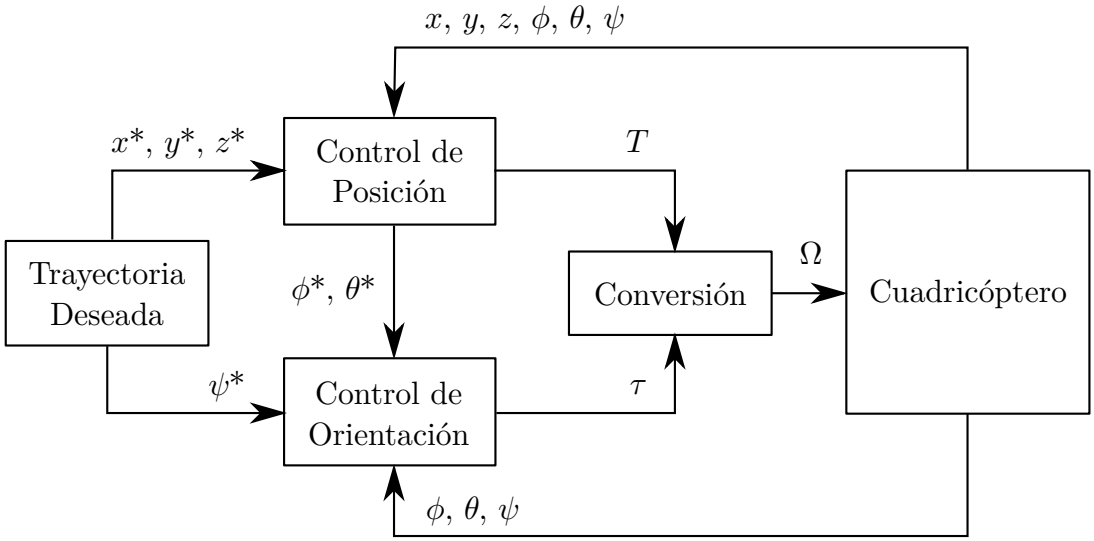


Figura 3.3 Diagrama para el control de trayectorias en un cuadricóptero

En la Figura 3.3 se ilustra el esquema del control para el modo de simulación. En primera instancia los valores de posición del cuadricóptero se comparan con los deseados de la trayectoria dentro del controlador de posición, para producir una aceleración lineal. Esta última en conjunto con el ángulo Yaw del cuadricóptero, se transforman con las expresiones (3.9) y (3.10), en los valores de ángulo Roll y Pitch deseados para el control de orientación. Ambas expresiones se obtienen al despejar dichos ángulos de (2.10) que considera sólo las fuerzas más relevantes. Igualmente por seguridad, estos valores se restringen por la expresión (3.11).

$$\phi^* = \arctan\left(\frac{\ddot{x}\sin\psi - \ddot{y}\cos\psi}{\sqrt{\left(\ddot{x}\cos\psi + \ddot{y}\sin\psi\right)^2 + \left(\ddot{z}+g\right)^2}}\right) \tag{3.9}$$

$$\theta^* = \arctan\left(\frac{\ddot{x}\cos\psi + \ddot{y}\sin\psi}{\ddot{z}+g}\right) \tag{3.10}$$

$$\begin{aligned} \phi_{min} \leq \phi^* \leq \phi_{max} \\ \theta_{min} \leq \theta^* \leq \theta_{max} \end{aligned} \tag{3.11}$$

Posteriormente, es el turno de comparar los valores anteriores y el ángulo Yaw deseado de la trayectoria con los valores de orientación del cuadricóptero en el controlador de orientación, generando así la aceleración angular. Con ambas aceleraciones se puede calcular entonces la señal de control para el cuadricóptero que corresponde a las velocidades de los rotores. Antes de ello, se debe calcular los valores de esfuerzo y torque necesarios con (3.12). Estas expresiones se obtienen del modelo del cuadricóptero simplificado que descarta los efectos complejos, como los centrípetos, los giroscópicos y el aire, agrupándolos como perturbaciones que deben corregir los controladores.

$$\begin{aligned} T_z &= \frac{m\left(\ddot{z}+g\right)}{\cos\phi\cos\theta} \\ \tau &= J\ddot{\eta} \end{aligned} \tag{3.12}$$

En este punto hace falta un último saturador, ya que los rotores sólo pueden producir un empuje y un torque limitado por las velocidades máximas y mínimas de éstos. Lo lógico sería estipular los límites por las desigualdades (3.13) que surgen de las ecuaciones (3.4). Sin embargo, un estudio a profundidad de éstas condujo a una serie de expresiones más específicas, seis sistemas de desigualdades (3.14), que evidencian la dependencia entre las variables esfuerzo y torque que no se observaba en (3.13).

$$\begin{aligned} 4k_l\Omega_{min}^2 \leq T_z \leq 4k_l\Omega_{max}^2 \\ |\tau_\phi| \leq \sqrt{2}lk_l\left(\Omega_{max}^2-\Omega_{min}^2\right) \\ |\tau_\theta| \leq \sqrt{2}lk_l\left(\Omega_{max}^2-\Omega_{min}^2\right) \\ |\tau_\psi| \leq 2k_d\left(\Omega_{max}^2-\Omega_{min}^2\right) \end{aligned} \tag{3.13}$$

$$\begin{cases} \Omega_{min}^2 \leq \frac{1}{4k_l}T_z + \frac{\sqrt{2}}{4lk_l}\tau_\phi \leq \Omega_{max}^2 \\ \Omega_{min}^2 \leq \frac{1}{4k_l}T_z - \frac{\sqrt{2}}{4lk_l}\tau_\phi \leq \Omega_{max}^2 \end{cases} \qquad \begin{cases} \Omega_{min}^2 \leq \frac{1}{4k_l}T_z + \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \Omega_{max}^2 \\ \Omega_{min}^2 \leq \frac{1}{4k_l}T_z - \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \Omega_{max}^2 \end{cases}$$

$$\begin{cases} \Omega_{min}^2 \leq \frac{1}{4k_l}T_z + \frac{1}{4k_d}\tau_\psi \leq \Omega_{max}^2 \\ \Omega_{min}^2 \leq \frac{1}{4k_l}T_z - \frac{1}{4k_d}\tau_\psi \leq \Omega_{max}^2 \end{cases} \qquad \begin{cases} -\frac{1}{2}\Omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \frac{1}{2}\Omega_{dif}^2 \\ -\frac{1}{2}\Omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \frac{1}{2}\Omega_{dif}^2 \end{cases} \tag{3.14}$$

$$\begin{cases} -\frac{1}{2}\Omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\Omega_{dif}^2 \\ -\frac{1}{2}\Omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\Omega_{dif}^2 \end{cases} \qquad \begin{cases} -\frac{1}{2}\Omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\theta + \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\Omega_{dif}^2 \\ -\frac{1}{2}\Omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\theta - \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\Omega_{dif}^2 \end{cases}$$

donde $\Omega_{dif}^2 = \Omega_{max}^2 - \Omega_{min}^2$

El estudio de cada conjunto de desigualdades se puede observar en las Tablas (3.2) y (3.3), en donde aparece también qué expresión se usa para su solución. Una vez pasado por estas restricciones, solo queda obtener los valores de velocidad de los rotores que servirán de entrada al modelo. La conversión se hace posible gracias a (3.15), que provienen de las ecuaciones (3.4).

$$\begin{aligned}
\Omega_1^2 &= \frac{1}{4k_l}T_z - \frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{\sqrt{2}}{4lk_l}\tau_\theta - \frac{1}{4k_d}\tau_\psi \\
\Omega_2^2 &= \frac{1}{4k_l}T_z - \frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{\sqrt{2}}{4lk_l}\tau_\theta + \frac{1}{4k_d}\tau_\psi \\
\Omega_3^2 &= \frac{1}{4k_l}T_z + \frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{\sqrt{2}}{4lk_l}\tau_\theta - \frac{1}{4k_d}\tau_\psi \\
\Omega_4^2 &= \frac{1}{4k_l}T_z + \frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{\sqrt{2}}{4lk_l}\tau_\theta + \frac{1}{4k_d}\tau_\psi
\end{aligned} \tag{3.15}$$

Por otro lado, el modo de implementación tiene un esquema de control más sencillo, que se puede ver en la Figura 3.4. Dicho modo se diferencia del simulado, en que la salida de los controladores se transforman en las señales de control por medio de las expresiones (3.16). Nótese que las señales relacionadas a los ángulos Roll y Pitch se pasan directamente al AR.Drone, debido a que sus controles se encuentran integrados en el mismo dispositivo.

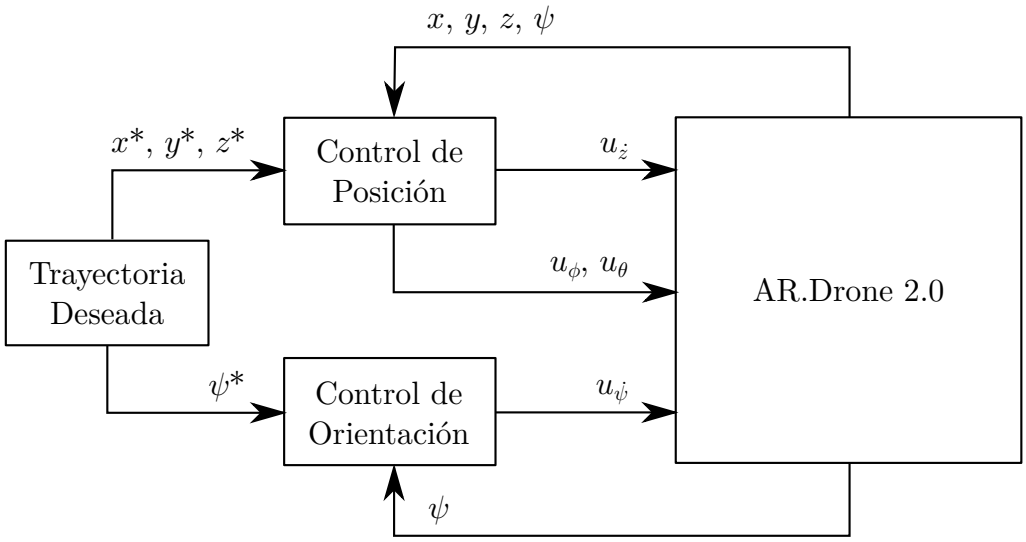


Figura 3.4 Diagrama para el control de trayectorias en el AR.Drone 2.0

Tabla 3.2 Restricciones de empuje y torque

$\omega_{min}^2 \leq \frac{1}{4k_l}T + \frac{\sqrt{2}}{4lk_l}\tau_\phi \leq \omega_{max}^2$ $\omega_{min}^2 \leq \frac{1}{4k_l}T - \frac{\sqrt{2}}{4lk_l}\tau_\phi \leq \omega_{max}^2$	$\frac{1}{4k_l}T - \frac{\sqrt{2}}{4lk_l}\tau_\phi < \omega_{min}^2$	$\omega_{min}^2 \leq \frac{1}{4k_l}T - \frac{\sqrt{2}}{4lk_l}\tau_\phi \leq \omega_{max}^2$	$\frac{1}{4k_l}T - \frac{\sqrt{2}}{4lk_l}\tau_\phi > \omega_{max}^2$
$\frac{1}{4k_l}T + \frac{\sqrt{2}}{4lk_l}\tau_\phi < \omega_{min}^2$	$T' = 4k_l\omega_{min}^2$ $\tau'_\phi = 0$	$T' = 2k_l\omega_{min}^2 + \frac{1}{2}T - \frac{\sqrt{2}}{2l}\tau_\phi$ $\tau'_\phi = \sqrt{2}lk_l\omega_{min}^2 - \frac{\sqrt{2}l}{4}T + \frac{1}{2}\tau_\phi$	$T' = 2k_l\omega_{sum}^2$ $\tau'_\phi = -\sqrt{2}lk_l\omega_{dif}^2$
$\omega_{min}^2 \leq \frac{1}{4k_l}T + \frac{\sqrt{2}}{4lk_l}\tau_\phi \leq \omega_{max}^2$	$T' = \frac{1}{2}T + \frac{\sqrt{2}}{2l}\tau_\phi + 2k_l\omega_{min}^2$ $\tau'_\phi = \frac{\sqrt{2}l}{4}T + \frac{1}{2}\tau_\phi - \sqrt{2}lk_l\omega_{min}^2$	$T' = T$ $\tau'_\phi = \tau_\phi$	$T' = \frac{1}{2}T + \frac{\sqrt{2}}{2l}\tau_\phi + 2k_l\omega_{max}^2$ $\tau'_\phi = \frac{\sqrt{2}l}{4}T + \frac{1}{2}\tau_\phi - \sqrt{2}lk_l\omega_{max}^2$
$\frac{1}{4k_l}T + \frac{\sqrt{2}}{4lk_l}\tau_\phi > \omega_{max}^2$	$T' = 2k_l\omega_{sum}^2$ $\tau'_\phi = \sqrt{2}lk_l\omega_{dif}^2$	$T' = 2k_l\omega_{max}^2 + \frac{1}{2}T - \frac{\sqrt{2}}{2l}\tau_\phi$ $\tau'_\phi = \sqrt{2}lk_l\omega_{max}^2 - \frac{\sqrt{2}l}{4}T + \frac{1}{2}\tau_\phi$	$T' = 4k_l\omega_{max}^2$ $\tau'_\phi = 0$
$\omega_{min}^2 \leq \frac{1}{4k_l}T + \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \omega_{max}^2$ $\omega_{min}^2 \leq \frac{1}{4k_l}T - \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \omega_{max}^2$	$\frac{1}{4k_l}T - \frac{\sqrt{2}}{4lk_l}\tau_\theta < \omega_{min}^2$	$\omega_{min}^2 \leq \frac{1}{4k_l}T - \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \omega_{max}^2$	$\frac{1}{4k_l}T - \frac{\sqrt{2}}{4lk_l}\tau_\theta > \omega_{max}^2$
$\frac{1}{4k_l}T + \frac{\sqrt{2}}{4lk_l}\tau_\theta < \omega_{min}^2$	$T' = 4k_l\omega_{min}^2$ $\tau'_\theta = 0$	$T' = 2k_l\omega_{min}^2 + \frac{1}{2}T - \frac{\sqrt{2}}{2l}\tau_\theta$ $\tau'_\theta = \sqrt{2}lk_l\omega_{min}^2 - \frac{\sqrt{2}l}{4}T + \frac{1}{2}\tau_\theta$	$T' = 2k_l\omega_{sum}^2$ $\tau'_\theta = -\sqrt{2}lk_l\omega_{dif}^2$
$\omega_{min}^2 \leq \frac{1}{4k_l}T + \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \omega_{max}^2$	$T' = \frac{1}{2}T + \frac{\sqrt{2}}{2l}\tau_\theta + 2k_l\omega_{min}^2$ $\tau'_\theta = \frac{\sqrt{2}l}{4}T + \frac{1}{2}\tau_\theta - \sqrt{2}lk_l\omega_{min}^2$	$T' = T$ $\tau'_\theta = \tau_\theta$	$T' = \frac{1}{2}T + \frac{\sqrt{2}}{2l}\tau_\theta + 2k_l\omega_{max}^2$ $\tau'_\theta = \frac{\sqrt{2}l}{4}T + \frac{1}{2}\tau_\theta - \sqrt{2}lk_l\omega_{max}^2$
$\frac{1}{4k_l}T + \frac{\sqrt{2}}{4lk_l}\tau_\theta > \omega_{max}^2$	$T' = 2k_l\omega_{sum}^2$ $\tau'_\theta = \sqrt{2}lk_l\omega_{dif}^2$	$T' = 2k_l\omega_{max}^2 + \frac{1}{2}T - \frac{\sqrt{2}}{2l}\tau_\theta$ $\tau'_\theta = \sqrt{2}lk_l\omega_{max}^2 - \frac{\sqrt{2}l}{4}T + \frac{1}{2}\tau_\theta$	$T' = 4k_l\omega_{max}^2$ $\tau'_\theta = 0$
$\omega_{min}^2 \leq \frac{1}{4k_l}T + \frac{1}{4k_d}\tau_\psi \leq \omega_{max}^2$ $\omega_{min}^2 \leq \frac{1}{4k_l}T - \frac{1}{4k_d}\tau_\psi \leq \omega_{max}^2$	$\frac{1}{4k_l}T - \frac{1}{4k_d}\tau_\psi < \omega_{min}^2$	$\omega_{min}^2 \leq \frac{1}{4k_l}T - \frac{1}{4k_d}\tau_\psi \leq \omega_{max}^2$	$\frac{1}{4k_l}T - \frac{1}{4k_d}\tau_\psi > \omega_{max}^2$
$\frac{1}{4k_l}T + \frac{1}{4k_d}\tau_\psi < \omega_{min}^2$	$T' = 4k_l\omega_{min}^2$ $\tau'_\psi = 0$	$T' = 2k_l\omega_{min}^2 + \frac{1}{2}T - \frac{k_l}{2k_d}\tau_\psi$ $\tau'_\psi = 2k_d\omega_{min}^2 - \frac{k_d}{2k_l}T + \frac{1}{2}\tau_\psi$	$T' = 2k_l\omega_{sum}^2$ $\tau'_\psi = -2k_d\omega_{dif}^2$
$\omega_{min}^2 \leq \frac{1}{4k_l}T + \frac{1}{4k_d}\tau_\psi \leq \omega_{max}^2$	$T' = \frac{1}{2}T + \frac{k_l}{2k_d}\tau_\psi + 2k_l\omega_{min}^2$ $\tau'_\psi = \frac{k_d}{2k_l}T + \frac{1}{2}\tau_\psi - 2k_d\omega_{min}^2$	$T' = T$ $\tau'_\psi = \tau_\psi$	$T' = \frac{1}{2}T + \frac{k_l}{2k_d}\tau_\psi + 2k_l\omega_{max}^2$ $\tau'_\psi = \frac{k_d}{2k_l}T + \frac{1}{2}\tau_\psi - 2k_d\omega_{max}^2$
$\frac{1}{4k_l}T + \frac{1}{4k_d}\tau_\psi > \omega_{max}^2$	$T' = 2k_l\omega_{sum}^2$ $\tau'_\psi = 2k_d\omega_{dif}^2$	$T' = 2k_l\omega_{max}^2 + \frac{1}{2}T - \frac{k_l}{2k_d}\tau_\psi$ $\tau'_\psi = 2k_d\omega_{max}^2 - \frac{k_d}{2k_l}T + \frac{1}{2}\tau_\psi$	$T' = 4k_l\omega_{max}^2$ $\tau'_\psi = 0$

Nota: $\omega_{sum}^2 = \omega_{max}^2 + \omega_{min}^2$ y $\omega_{dif}^2 = \omega_{max}^2 - \omega_{min}^2$

Tabla 3.3 Restricciones de empuje y torque (continuación)

$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \frac{1}{2}\omega_{dif}^2$ $-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \frac{1}{2}\omega_{dif}^2$	$\frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{\sqrt{2}}{4lk_l}\tau_\theta < -\frac{1}{2}\omega_{dif}^2$	$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \frac{1}{2}\omega_{dif}^2$	$\frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{\sqrt{2}}{4lk_l}\tau_\theta > \frac{1}{2}\omega_{dif}^2$
$\frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{\sqrt{2}}{4lk_l}\tau_\theta < -\frac{1}{2}\omega_{dif}^2$	$\tau'_\phi = -\sqrt{2}lk_l\omega_{dif}^2$ $\tau'_\theta = 0$	$\tau'_\phi = -\frac{\sqrt{2}lk_l}{2}\omega_{dif}^2 + \frac{1}{2}\tau_\phi - \frac{1}{2}\tau_\theta$ $\tau'_\theta = -\frac{\sqrt{2}lk_l}{2}\omega_{dif}^2 - \frac{1}{2}\tau_\phi + \frac{1}{2}\tau_\theta$	$\tau'_\phi = 0$ $\tau'_\theta = -\sqrt{2}lk_l\omega_{dif}^2$
$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{\sqrt{2}}{4lk_l}\tau_\theta \leq \frac{1}{2}\omega_{dif}^2$	$\tau'_\phi = \frac{1}{2}\tau_\phi + \frac{1}{2}\tau_\theta - \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2$ $\tau'_\theta = \frac{1}{2}\tau_\phi + \frac{1}{2}\tau_\theta + \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2$	$\tau'_\phi = \tau_\phi$ $\tau'_\theta = \tau_\theta$	$\tau'_\phi = \frac{1}{2}\tau_\phi + \frac{1}{2}\tau_\theta + \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2$ $\tau'_\theta = \frac{1}{2}\tau_\phi + \frac{1}{2}\tau_\theta - \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2$
$\frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{\sqrt{2}}{4lk_l}\tau_\theta > \frac{1}{2}\omega_{dif}^2$	$\tau'_\phi = 0$ $\tau'_\theta = \sqrt{2}lk_l\omega_{dif}^2$	$\tau'_\phi = \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2 + \frac{1}{2}\tau_\phi - \frac{1}{2}\tau_\theta$ $\tau'_\theta = \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2 - \frac{1}{2}\tau_\phi + \frac{1}{2}\tau_\theta$	$\tau'_\phi = \sqrt{2}lk_l\omega_{dif}^2$ $\tau'_\theta = 0$
$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\omega_{dif}^2$ $-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\omega_{dif}^2$	$\frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{1}{4k_d}\tau_\psi < -\frac{1}{2}\omega_{dif}^2$	$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\omega_{dif}^2$	$\frac{\sqrt{2}}{4lk_l}\tau_\phi - \frac{1}{4k_d}\tau_\psi > \frac{1}{2}\omega_{dif}^2$
$\frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{1}{4k_d}\tau_\psi < -\frac{1}{2}\omega_{dif}^2$	$\tau'_\phi = -\sqrt{2}lk_l\omega_{dif}^2$ $\tau'_\psi = 0$	$\tau'_\phi = -\frac{\sqrt{2}lk_l}{2}\omega_{dif}^2 + \frac{1}{2}\tau_\phi - \frac{\sqrt{2}lk_l}{4k_d}\tau_\psi$ $\tau'_\psi = -k_d\omega_{dif}^2 - \frac{\sqrt{2}k_d}{2lk_l}\tau_\phi + \frac{1}{2}\tau_\psi$	$\tau'_\phi = 0$ $\tau'_\psi = -2k_d\omega_{dif}^2$
$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\omega_{dif}^2$	$\tau'_\phi = \frac{1}{2}\tau_\phi + \frac{\sqrt{2}lk_l}{4k_d}\tau_\psi - \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2$ $\tau'_\psi = \frac{\sqrt{2}k_d}{2lk_l}\tau_\phi + \frac{1}{2}\tau_\psi + k_d\omega_{dif}^2$	$\tau'_\phi = \tau_\phi$ $\tau'_\psi = \tau_\psi$	$\tau'_\phi = \frac{1}{2}\tau_\phi + \frac{\sqrt{2}lk_l}{4k_d}\tau_\psi + \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2$ $\tau'_\psi = \frac{\sqrt{2}k_d}{2lk_l}\tau_\phi + \frac{1}{2}\tau_\psi - k_d\omega_{dif}^2$
$\frac{\sqrt{2}}{4lk_l}\tau_\phi + \frac{1}{4k_d}\tau_\psi > \frac{1}{2}\omega_{dif}^2$	$\tau'_\phi = 0$ $\tau'_\psi = 2k_d\omega_{dif}^2$	$\tau'_\phi = \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2 + \frac{1}{2}\tau_\phi - \frac{\sqrt{2}lk_l}{4k_d}\tau_\psi$ $\tau'_\psi = k_d\omega_{dif}^2 - \frac{\sqrt{2}k_d}{2lk_l}\tau_\phi + \frac{1}{2}\tau_\psi$	$\tau'_\phi = \sqrt{2}lk_l\omega_{dif}^2$ $\tau'_\psi = 0$
$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\theta + \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\omega_{dif}^2$ $-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\theta - \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\omega_{dif}^2$	$\frac{\sqrt{2}}{4lk_l}\tau_\theta - \frac{1}{4k_d}\tau_\psi < -\frac{1}{2}\omega_{dif}^2$	$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\theta - \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\omega_{dif}^2$	$\frac{\sqrt{2}}{4lk_l}\tau_\theta - \frac{1}{4k_d}\tau_\psi > \frac{1}{2}\omega_{dif}^2$
$\frac{\sqrt{2}}{4lk_l}\tau_\theta + \frac{1}{4k_d}\tau_\psi < -\frac{1}{2}\omega_{dif}^2$	$\tau'_\theta = -\sqrt{2}lk_l\omega_{dif}^2$ $\tau'_\psi = 0$	$\tau'_\theta = -\frac{\sqrt{2}lk_l}{2}\omega_{dif}^2 + \frac{1}{2}\tau_\theta - \frac{\sqrt{2}lk_l}{4k_d}\tau_\psi$ $\tau'_\psi = -k_d\omega_{dif}^2 - \frac{\sqrt{2}k_d}{2lk_l}\tau_\theta + \frac{1}{2}\tau_\psi$	$\tau'_\theta = 0$ $\tau'_\psi = -2k_d\omega_{dif}^2$
$-\frac{1}{2}\omega_{dif}^2 \leq \frac{\sqrt{2}}{4lk_l}\tau_\theta + \frac{1}{4k_d}\tau_\psi \leq \frac{1}{2}\omega_{dif}^2$	$\tau'_\theta = \frac{1}{2}\tau_\theta + \frac{\sqrt{2}lk_l}{4k_d}\tau_\psi - \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2$ $\tau'_\psi = \frac{\sqrt{2}k_d}{2lk_l}\tau_\theta + \frac{1}{2}\tau_\psi + k_d\omega_{dif}^2$	$\tau'_\theta = \tau_\theta$ $\tau'_\psi = \tau_\psi$	$\tau'_\theta = \frac{1}{2}\tau_\theta + \frac{\sqrt{2}lk_l}{4k_d}\tau_\psi + \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2$ $\tau'_\psi = \frac{\sqrt{2}k_d}{2lk_l}\tau_\theta + \frac{1}{2}\tau_\psi - k_d\omega_{dif}^2$
$\frac{\sqrt{2}}{4lk_l}\tau_\theta + \frac{1}{4k_d}\tau_\psi > \frac{1}{2}\omega_{dif}^2$	$\tau'_\theta = 0$ $\tau'_\psi = 2k_d\omega_{dif}^2$	$\tau'_\theta = \frac{\sqrt{2}lk_l}{2}\omega_{dif}^2 + \frac{1}{2}\tau_\theta - \frac{\sqrt{2}lk_l}{4k_d}\tau_\psi$ $\tau'_\psi = k_d\omega_{dif}^2 - \frac{\sqrt{2}k_d}{2lk_l}\tau_\theta + \frac{1}{2}\tau_\psi$	$\tau'_\theta = \sqrt{2}lk_l\omega_{dif}^2$ $\tau'_\psi = 0$

Nota: $\omega_{dif}^2 = \omega_{max}^2 - \omega_{min}^2$

$$
\begin{aligned}
u_{\phi} &= \frac{\phi^*}{\phi_{max}} & u_{\theta} &= \frac{\theta^*}{\theta_{max}} \\
u_{\dot{z}} &= \frac{\Delta t \ \ddot{z}}{\dot{z}_{max}} & u_{\dot{\psi}} &= \frac{\Delta t \ \ddot{\psi}}{\dot{\psi}_{max}}
\end{aligned}
\tag{3.16}
$$

Una vez aclarado estos aspectos del lazo de control, sólo queda enfocarse en el controlador. Dado que el estudio se basa en poner a prueba un control inteligente frente a un control convencional, la elección de los controladores asociados es crucial para el rendimiento que demuestren. Por el lado del control inteligente, un controlador difuso es la propuesta que se plantea, mientras que por el lado del control convencional, un controlador PID es la opción más factible.

En relación al controlador difuso, se tiene que su metodología de diseño aplica esencialmente heurística y ciertos principios de la Inteligencia Artificial. El diseñador debe incorporarle conocimiento experto sobre como resolver el problema de control. También se requiere establecer varios parámetros de ajuste para conseguir un buen rendimiento, aunque esto no es una tarea sencilla.

Para empezar, las entradas de los controladores difusos, tanto de posición como de orientación, se definen como el error y la derivada del error de las variables de interés, mientras que su salida corresponden a la aceleración lineal o angular según el caso.

Como se puede observar en la Figura 3.5, ambas entradas usan los mismos tres conjuntos difusos: N (*Negative*), Z (*Zero*), P (*Positive*), cosa diferente para la salida que utiliza cinco conjuntos difusos: VN (*Very-Negative*), N (*Negative*), Z (*Zero*), P (*Positive*), VP (*Very-Positive*). Además, por simplicidad todos tienen forma triangular a excepción de los extremos que se encuentran truncados. También se establece un rango fijo normalizado para las variables de los conjuntos difusos, y se asocia un parámetro de ganancia para la conversión en cada entrada (K_p, K_d) y salida (K_o), con el objetivo de reducir el número de parámetros involucrados a sólo estos tres. El ajuste de los mismos se puede hacer de manera empírica o utilizando técnicas de optimización.

Asimismo, la base de reglas suministrada se muestra en la Tabla 3.4, en donde las reglas difusas de la forma SI<antecedente>,ENTONCES<consecuente>están codificadas en un formato de matriz asociativa difusa. Como dato curioso está la simetría diagonal de dicha matriz, debido a la semejanza de acción en todas las direcciones.

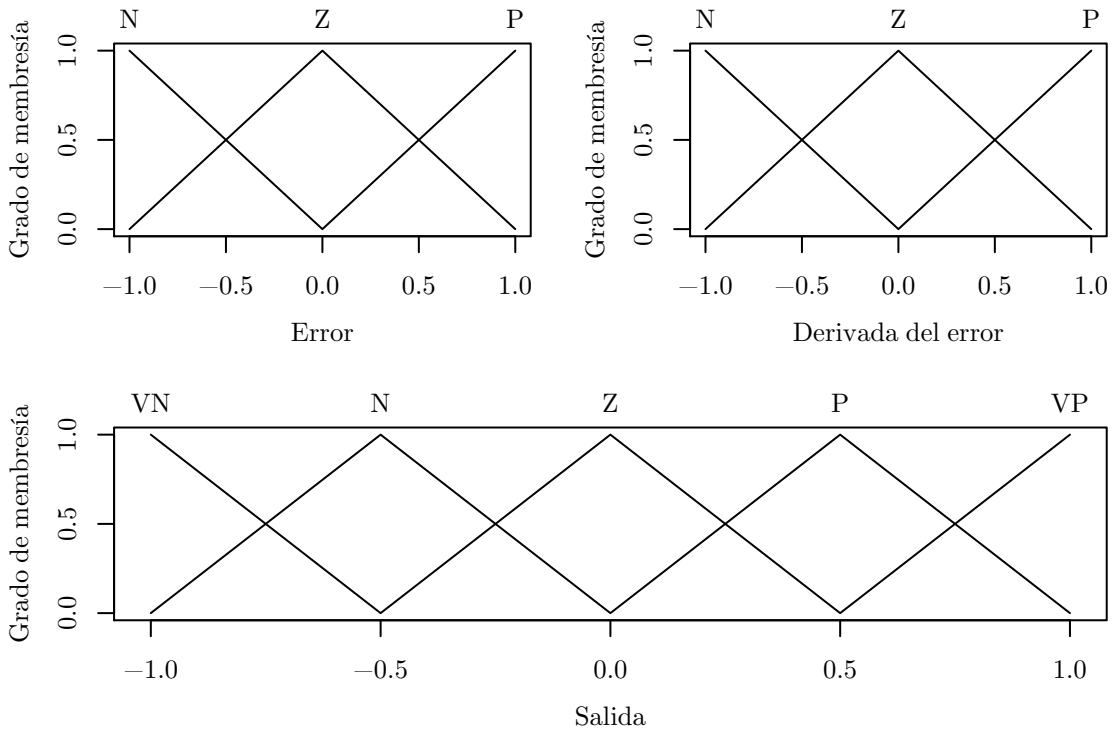


Figura 3.5 Funciones de membresía para el controlador difuso

Otro aspecto importante es el mecanismo de inferencia que utiliza el controlador, en concreto de tipo Mamdani, principalmente por ser el más utilizado en el estado del arte. Esta elección conlleva a que el operador mínimo se aplique en la conjunción (AND) y en la implicación difusa, el operador máximo en la unión (OR) y el operador máx-mín en la composición de inferencia difusa. También hay que aclarar que el método designado para calcular la salida combinada del controlador difuso es el del centroide, debido al buen desempeño que ha demostrado en trabajos relacionados. Todas las operaciones nombradas están asociadas a la lógica difusa y no tienen parámetros de ajuste.

Tabla 3.4 Base de reglas difusas

Error \ Derivada del error	N	Z	P
N	VN	N	Z
Z	N	Z	P
P	Z	P	VP

Con respecto al controlador PID, se puede acotar que su diseño es simple debido a la expresión que lo define (3.17), en donde $e\,(t)$ es el error de entrada de la variable de interés, y $\ddot{q}\,(t)$ es la salida de control correspondiente a la aceleración lineal o angular según el caso. Tal como se observa los únicos parámetros de ajuste son K_p, K_i y K_d, pero establecer los valores óptimos no es tarea sencilla ya que sus aportes muchas veces entran en conflicto.

$$\ddot{q}\,(t) = K_p e\,(t) + K_i \int e\,(t)\ \ dt + K_d \frac{de\,(t)}{dt} \tag{3.17}$$

Una práctica muy común para calcular estos valores es valerse de métodos basados en tablas como el Ziegler-Nichols, el Cohen-Coon, el Chien-Hrones-Reswick y el Astrom-Hagglund. Éstos pueden aplicarse con solo tener un modelo del proceso, pero desafortunadamente están diseñados para controles tipo regulador, es decir, aquellos donde el valor deseado se mantiene fijo o varia muy lentamente respecto al control. Sin embargo, dicha premisa es una limitación para los objetivos del estudio.

Algunos autores atacan el problema realizando un estudio matemático muy completo del control sobre el modelo, pero existe un requisito para este enfoque, tener definido con antelación las trayectorias a realizar ya que es necesario incorporarlas en los cálculos. El presente estudio busca justamente tener la libertad de seleccionar las trayectorias al momento de establecer las pruebas.

Otros estudios simplemente definen las ganancias de manera empírica en función de las pruebas a realizar, o también con alguna técnica de optimización en caso de estar disponible. Esta opción parece la más congruente teniendo en cuenta que los parámetros del controlador difuso se definen tal cual, y para ser justos en los criterios de comparación, el ajuste de sus parámetros no deben diferir.

Por último y no menos importante, en la programación del controlador PID únicamente fue necesaria la ecuación (3.17), mientras que para el controlador difuso se necesitó de una librería externa, FuzzyLogic, que suministra toda la parte de la lógica difusa [74]. El tipo de controlador a utilizar se define directamente desde un archivo de configuración sin la necesidad de recompilar para facilidad.

3.5 Módulo Supervisor

Hace falta un ente que se encargue de velar por el buen funcionamiento del sistema. El supervisor representa la autoridad maestra que rige sobre las diversas áreas, siendo a su vez el factor de acción cuando se presenta un problema. El primer aspecto a destacar es que él administra los controles de ejecución de la prueba, da la orden de inicio, de fin, y también en el caso de la simulación, lo opción de pausar y reanudar. Todas estas acciones son estipuladas por el usuario por medio del teclado.

Una de las funciones más importantes que tiene es llevar el seguimiento de la variable tiempo t necesaria tanto para la trayectoria como para lo relacionado al modelo. Este tiempo se hace avanzar desde un tiempo inicial t_o hasta un tiempo final t_f en tamaños de paso Δt. Otra de sus virtudes es la posibilidad de ejecutar el algoritmo en tiempo real, es decir, que cada ciclo del algoritmo se efectúe en lapsos de tiempo Δt reales. Esto es posible siempre y cuando la velocidad del computador lo permita, y en caso de sobrepasar el tiempo, un mensaje de advertencia lo comunica por consola. La medición de tiempos de ejecución se logra a través de la librería estándar chrono, capaz de manejar medidas de microsegundos. Dicho aspecto no es relevante en el modo simulación pero sí es obligatorio para el de implementación.

Con respecto al AR.Drone, se requiere que antes de la prueba, se eleve hasta un punto de suspensión, y después de la misma, retorne al suelo. Este proceso es manejado por el supervisor, como también los casos de emergencia más comunes, que el vehículo informe de algún error o que exista problemas con la comunicación. En función de la gravedad puede optar a mantener el dispositivo en modo suspendido u ordenarle que aterrice por seguridad.

Por último, existe un modo especial de implementación, en donde el supervisor desconecta el control habitual, y establece un control manual del AR.Drone a través del teclado. Dicha función está enfocada para hacer pruebas directas sobre las señales de control y para recabar información sobre la posición y orientación. El modo especial se habilita durante la compilación del código.

3.6 Módulo de Comunicación

La incorporación de una comunicación es evidente para hacer compatible el sistema con el AR.Drone. En el capítulo anterior se mencionaron brevemente los protocolos involucrados, sin embargo, se requieren de varias instrucciones adicionales para entenderse con el cuadricóptero, en especial durante la inicialización. Es un proceso posible pero algo complejo para el que no está familiarizado con él, además que requiere invertir mucho tiempo y escasea la documentación técnica relacionada. Por las razones expuestas se decidió buscar otra alternativa.

Existe una colección de herramientas y librerías llamada ROS (*Robotic Operating System*), destinadas a simplificar la tarea de programar un comportamiento robótico complejo y robusto a través de una infinidad de plataformas robóticas. Su principal ventaja es que brinda un marco flexible para programar robots, por lo que se ha convertido en un estándar en el área de investigación de robótica.

Implementar la comunicación bajo este entorno de trabajo, facilita enormemente la tarea ya que todo se reduce a implementar una comunicación estándar de ROS, que además de ser sencillo, está muy bien documentado. Asimismo, trae el beneficio de que, en caso de necesitarse la incorporación de otras herramientas, no existan problemas de compatibilidad, aún si éstas fueran diseñadas para otro fin.

ROS tiene una arquitectura en forma de grafo donde todo el procesamiento lo realizan los nodos asociados. Cada nodo es independiente y se relacionan con el resto utilizando un modelo publicador/subscriptor. La comunicación entre nodos se realiza mediante el paso de mensajes asociados a un tópico o etiqueta. Existen también los servicios, otra modalidad de conexión entre nodos en los que se requiere una respuesta. La librería está orientada para sistemas Linux, aunque hay intentos de adaptarlo a otros sistemas.

La plataforma de interés, el AR.Drone, cuenta con su propio paquete en ROS llamado *ardrone_autonomy*, desarrollado por Mani Monajjemi [75, 76]. El paquete está basado en el SDK oficial del dispositivo, y ejerce el papel de intermediario entre la aplicación y el cuadricóptero. La documentación establece que se pueden enviar instrucciones sencillas de despegue y aterrizaje, así como también comandos

de movimiento, compuestos por las mismas señales de control del SDK [77]. Por el lado de la información que devuelve, se tiene una lista bastante extensa de variables, pero las que concierne a la investigación son el estado del cuadricóptero (Inicializando, Preparado, Despegando, Suspendido, Volando y Aterrizando), y la odometría. Esta última es muy importante ya que suministra una estimación de la posición y orientación del cuadricóptero a partir de sus sensores, evitando tener que calcularlo en el sistema.

Un paquete complementario llamado *ar_track_alvar*, también es requerido en el entorno de ROS [78, 79]. Dicho paquete se utiliza para el seguimiento de marcas o *tags* en el espacio tipo AR (*Augmented Reality*), algunas de las cuales se pueden observar en la Figura 3.6. Esta técnica calcula transformaciones proyectivas en base al tamaño y distorsión de las marcas en una imagen determinada. Con una cámara de buena resolución, como la frontal del AR.Drone, se pueden obtener imágenes que permitan estimar la posición y orientación de las marcas respecto a dicha cámara. Con estos datos, y conociendo de antemano la localización de las marcas en el marco global, se puede ubicar sin problema cualquier dispositivo.

Figura 3.6 Ejemplos de tags incluidos en el packete de ROS *ar_track_alvar*

Si bien ambos paquetes entregan una posición y una orientación del AR.Drone, los dos son necesarios para el funcionamiento del sistema, constituyendo la razón principal de la sección de estimación. Una vez establecidos los paquetes, el siguiente punto consistía en hacer que el sistema se entendiera con ellos, para lo cual se necesitaba que hablaran el mismo idioma. La solución vino de las librerías de ROS que definían la estructura de los mensajes. Incorporándolas al código, y asignando los tópicos de envío respectivos, fue suficiente para establecer correctamente la comunicación. La Tabla 3.5, resume toda la información relacionada a los tipos de mensajes y tópicos de los nodos.

Tabla 3.5 Mensajes y tópicos de comunicación con los nodos de ROS

Nodo <Tipo>	Mensaje <Tópico>	Descripción
ardrone_autonomy <Publicador>	*std_msgs::Empty* *<ardrone/takeoff>*	Orden de Despegar
ardrone_autonomy <Publicador>	*std_msgs::Empty* *<ardrone/land>*	Orden de Aterrizar
ardrone_autonomy <Publicador>	*geometry_msgs::Twist* *<cmd_vel>*	Comandos de Movimientos
ardrone_autonomy <Subscriptor>	*ardrone_autonomy::Navdata* *<ardrone/navdata>*	Estado del Dron
ardrone_autonomy <Subscriptor>	*nav_msgs::Odometry* *<ardrone/odometry>*	Posición y Orientación
ar_track_alvar <Subscriptor>	*ar_track_alvar::AlvarMarkers* *<ar_pose_marker>*	Posición y Orientación

Toda la comunicación entre el sistema y el AR.Drone, incluidos los nodos de los paquetes, se puede observar en la Figura 3.7. Dichos nodos se muestran en el mismo equipo del sistema aunque la arquitectura de ROS permite también ejecutarlos desde otros diferentes. Nótese que la entrada al nodo *ar_track_alvar* es proporcionada por el nodo *ardrone_autonomy* que se encarga de toda la interacción con el AR.Drone. Cabe destacar que el modo implementación es el que incorpora toda la parte de ROS.

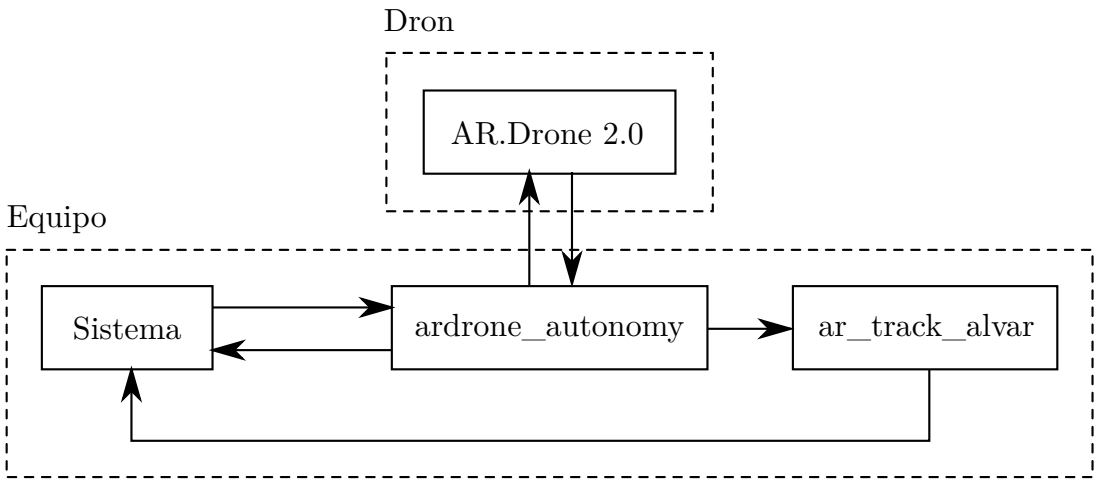


Figura 3.7 Esquema de comunicación entre el sistema y el AR.Drone 2.0

3.7 Módulo de Estimación

Se sabe que el AR.Drone tiene la capacidad de ejecutar el despegue y aterrizaje de manera autónoma, así como también mantenerse flotando y realizar ciertos movimientos simples. Sin embargo, una vez se encuentra en el aire, la realidad muestra que el cuadricóptero comienza a perder rendimiento. Si está en suspensión, poco a poco se aleja de su posición inicial, y si en cambio ejecuta un movimiento, lo hace de manera incompleta o agregando movimientos no deseados.

El efecto descrito se conoce en la literatura como *drifting* y se debe principalmente a errores de medición acumulados a lo largo del tiempo. En función de garantizar un mejor desempeño durante el vuelo estacionario, el posicionamiento o las maniobras de seguimiento de trayectoria, se requiere no sólo del control, sino también de una estimación continua de la posición y la orientación del AR.Drone a partir de referencias externas.

En la sección de comunicación, se habla de dos formas disponibles para obtener dicha información, la que es por odometría, que se basa en la medición de los sensores incorporados en el AR.Drone, y la que es por marcas externas, que se extraen y analizan de imágenes capturadas de la cámara frontal, ambas referidas como «Odo» y «Tag» para mayor facilidad. El uso de cada una tiene sus ventajas y desventajas como se explica a continuación.

«Odo» obtiene información continua y precisa a cada instante por el uso de sensores, lo cual es fundamental si existen movimientos ágiles, pero como ya se explicó, conlleva también acumulación de errores con el tiempo que afectan a la estimación. Por su parte, «Tag» emplea directamente una imagen que no depende de información anterior, lo cual le brinda un resultado bastante cercano al exacto, sin embargo el procesamiento de cada imagen requiere cierto tiempo, y muchas veces no es muy preciso, debido al ruido en la imagen producido por los movimientos bruscos inherentes a la dinámica del cuadricóptero.

Como se observa, existen deficiencias de manera individual, pero si se combinan las mejores características de cada una, se puede generar una muy buena estimación de la posición y la orientación. Tanto «Odo» como «Tag», representan la misma señal

obtenida por diferentes métodos, por lo que la idea es combinarlas eliminando de cada una, los aspectos desfavorables. En el caso de «Odo», el problema está asociado a las bajas frecuencias, mientras que en «Tag», lo es las altas frecuencias. Un filtro pasa alto para la primera y otro pasa bajo para la segunda, ajustados a la misma frecuencia, se planteó como solución.

Se optó por emplear filtros digitales tipo IIR (*Infinite Impulse Response*), que en comparación con sus homólogos FIR (*Finite Impulse Response*), tienen recursividad y por lo tanto no requieren tantas muestras pasadas para generar pendientes abruptas. Las ecuaciones de los filtros aplicados son (3.18), para el pasa alto y (3.19) para el pasa bajo, extraidas de [80]. Es evidente que se utiliza sólo una muestra anterior en el cálculo, y que el parámetro de ajuste α debe cumplir la condición $|\alpha| < 1$ para que el filtro sea estable.

$$H(z) = \frac{1+\alpha}{2}\left(\frac{1-z^{-1}}{1-\alpha z^{-1}}\right) \qquad |\alpha| < 1 \tag{3.18}$$

$$H(z) = \frac{1-\alpha}{2}\left(\frac{1+z^{-1}}{1-\alpha z^{-1}}\right) \qquad |\alpha| < 1 \tag{3.19}$$

Este parámetro α se relaciona con la frecuencia de corte digital $\hat{f}_c \in [-\frac{1}{2}, \frac{1}{2}]$ por medio de las expresiones (3.20) y (3.21) según [80].

$$\hat{f}_c = \frac{1}{2\pi}\arccos\left(\frac{2\alpha}{1+\alpha^2}\right) \tag{3.20}$$

$$\alpha = \frac{1-\sin\left(2\pi\hat{f}_c\right)}{\cos\left(2\pi\hat{f}_c\right)} \tag{3.21}$$

La relación entre frecuencias analógicas y digitales está dada por $\hat{f} = f/f_m$, en donde $\hat{f}$ es la frecuencia digital, f es la frecuencia analógica y f_m es la frecuencia de muestreo de la señal. Un punto importante es que esta última frecuencia debe cumplir el criterio de Nyquist, es decir, ser al menos el doble del ancho de banda de la señal $f_m > 2W$. Para el caso de estudio, el algoritmo efectúa una iteración en un tiempo Δt, con lo cual $f_m = 1/\Delta t$ y por ende se obtiene la restricción de implementación $\Delta t < 1/(2W)$.

El problema de la frecuencia digital es que requiere que se defina Δt, cosa que no ocurre hasta las pruebas, por lo que se prefiere dejar la expresión en términos de la frecuencia analógica. La ventaja de ello, es que hallado el punto de corte en frecuencias analógicas, será invariante con respecto a Δt. Por lo tanto, si f_c es constante, se puede hallar una relación entre α y Δt.

De la ecuación (3.21) se puede deducir la ecuación (3.22) que depende de Δt, y tiene una gráfica como la mostrada en la Figura 3.8, cuyo rango válido es para $\Delta t \geq 0$. Se puede observar que en la región donde Δt es cercano a cero, se tiene un comportamiento similar a una recta, lo que se puede confirmar si se calcula la pendiente en el punto $\Delta t = 0$ (3.23). Por lo tanto, un Δt lo suficientemente pequeño permite afirmar la aproximación (3.24) siendo válido sólo si $\Delta t < 0.15/(2\pi f_c)$, umbral que supera el 1 % de error.

$$\alpha\left(\Delta t\right) = \frac{1 - \sin\left(2\pi f_c \Delta t\right)}{\cos\left(2\pi f_c \Delta t\right)} \tag{3.22}$$

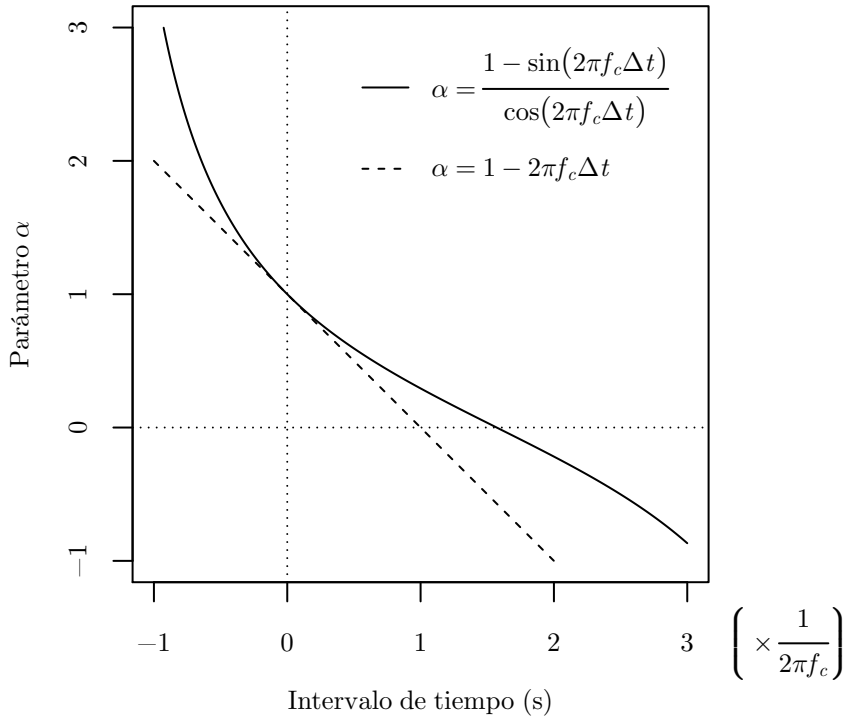


Figura 3.8 Curva del parámetro del filtro y la recta más aproxima

$$\dot{\alpha}(0) = (2\pi f_c) \left. \frac{\sin(2\pi f_c \Delta t) - 1}{\cos^2(2\pi f_c \Delta t)} \right|_{\Delta t = 0} = -2\pi f_c \tag{3.23}$$

$$\alpha(\Delta t) \approx 1 - 2\pi f_c \Delta t \tag{3.24}$$

La expresión deducida servirá siempre que se consiga el valor de f_c. Para calcularlo, se efectúa una barrido de α en ambos filtros a la vez, y se observa la señal resultante de la mezcla por adición, llamada «Mix» por conveniencia. El objetivo es que «Mix» contenga las altas frecuencias de «Odo» y las bajas frecuencias del «Tag», si se ve desde el punto de vista frecuencial, o que tenga un nivel continuo similar a «Tag», pero que los cambios violentos sean regidos por «Odo», si se ve desde el punto de vista temporal.

Con un barrido a pasos de 0.01, el valor que produjo la mejor señal combinada fue $\alpha = 0.98$ que corresponde a un $f_c = 0.1608$ Hz. En las Figuras 3.9 y 3.10 se observa claramente las características en tiempo y frecuencia de la señal «Mix», junto a las señales «Odo» y «Tag», que tanto se habían comentado con anterioridad.

Es posible apreciar en las gráficas de tiempo, como «Odo» parte de un nivel nulo ya que no tiene referencia alguna del espacio, mientras que «Tag» sí, pero este último demuestra un ruido y retardo tal que no tiene comparación con los de «Odo». Asimismo, en las gráficas de frecuencia se detalla el ancho de banda de las señales y donde se concentra su aporte más importante. Por donde se vea, la señal «Mix» es una solución inteligente que solventa muchos de los problemas inherentes.

Otras características complementarias en el análisis de señales son la media y la desviación estándar, las cuales se muestran en la Figura 3.11 para una ventana de 25 muestras. Allí se evidencia como la señal «Mix» comparte una media muy parecida a la señal de «Tag» pero una desviación estándar del orden de la señal de «Odo».

Los datos adquiridos para las tres figuras, provienen de una prueba real con el AR.Drone 2.0 en la que el control estaba inactivo y sólo se le indicó quedarse suspendido. Por lo que se aprecia, igualmente existen una dinámica elevada en este modo.

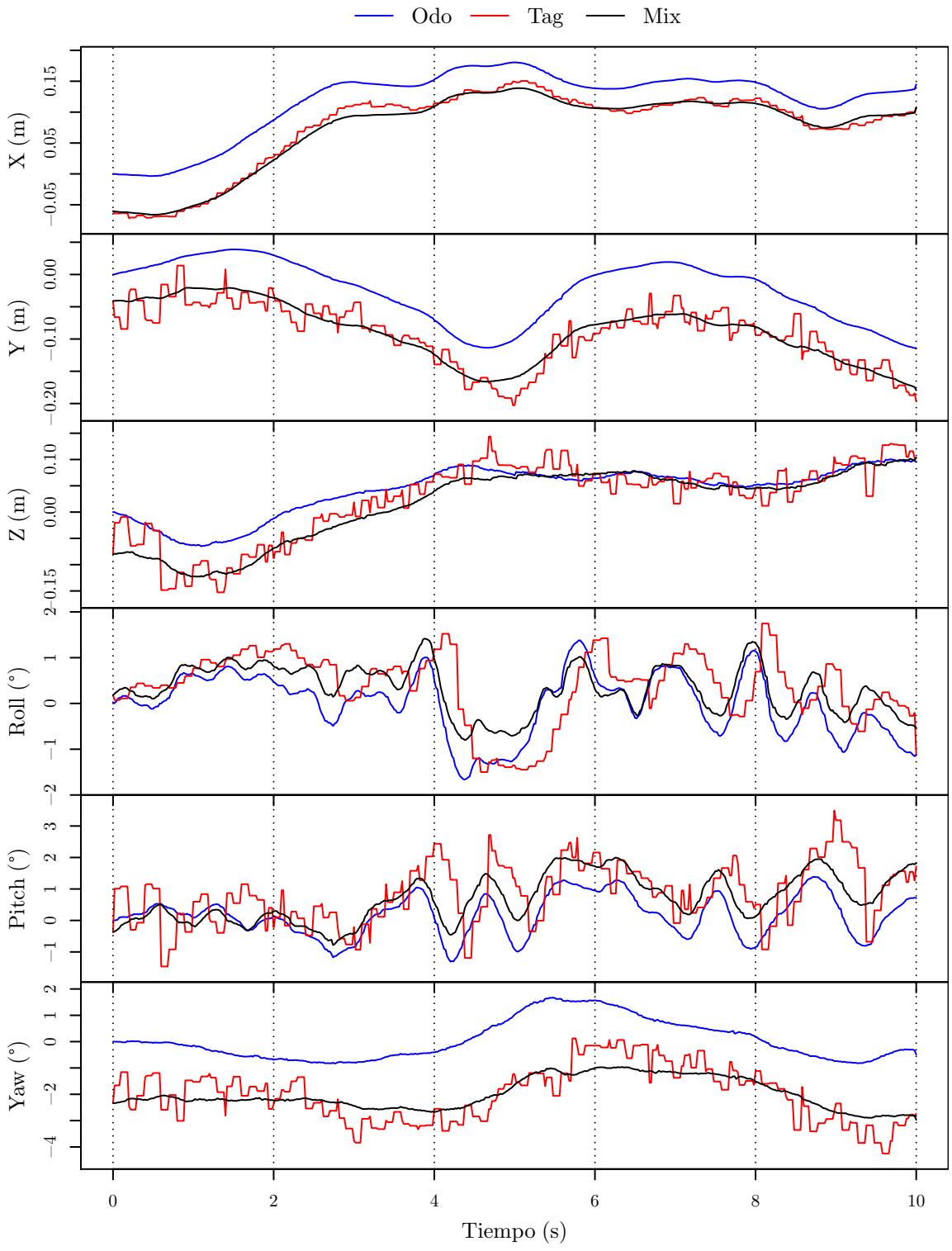


Figura 3.9 Valor temporal de la estimación de la posición y la orientación del AR.Drone 2.0 por medio de diferentes métodos

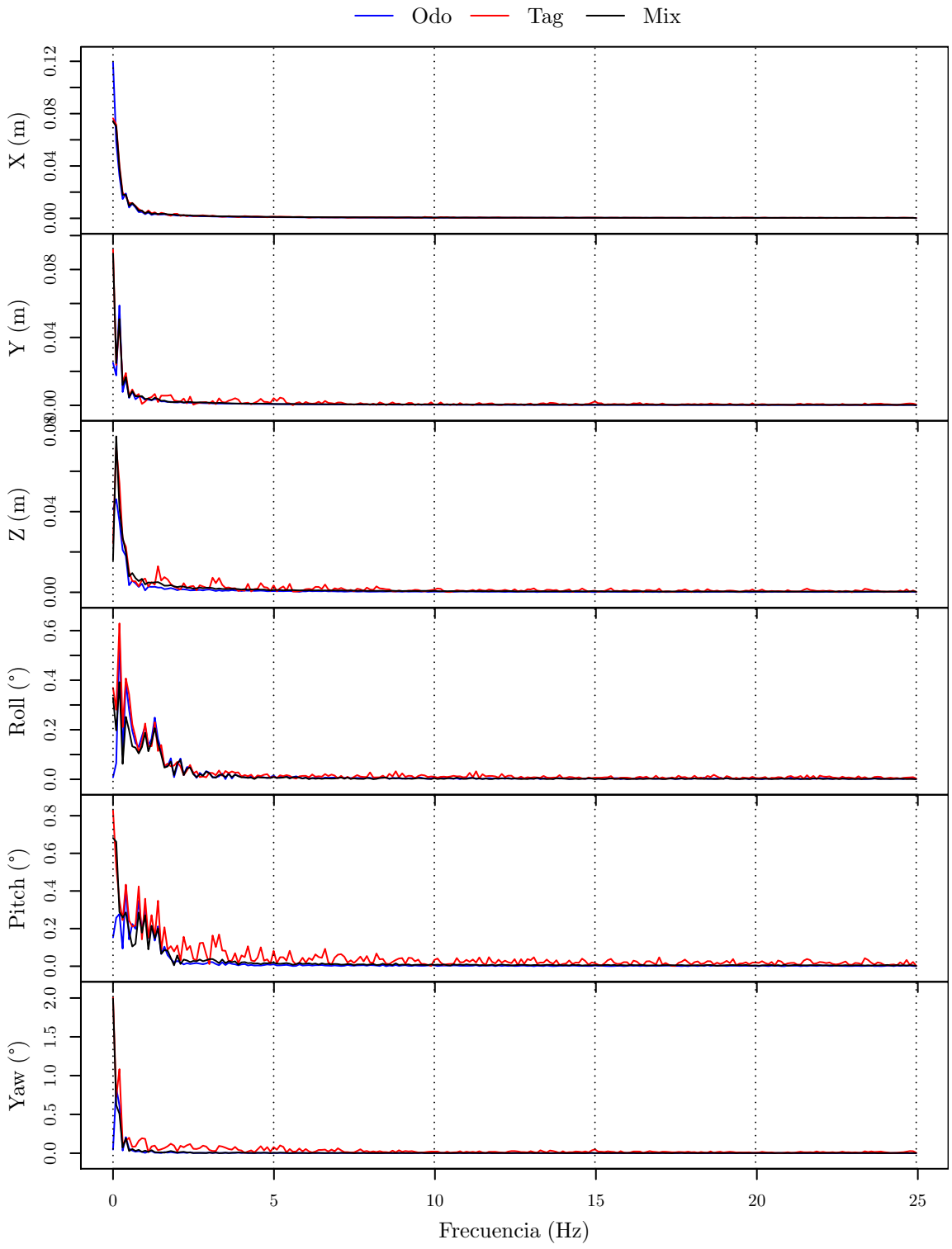


Figura 3.10 Contenido frecuencial de la estimación de la posición y la orientación del AR.Drone 2.0 por medio de diferentes métodos

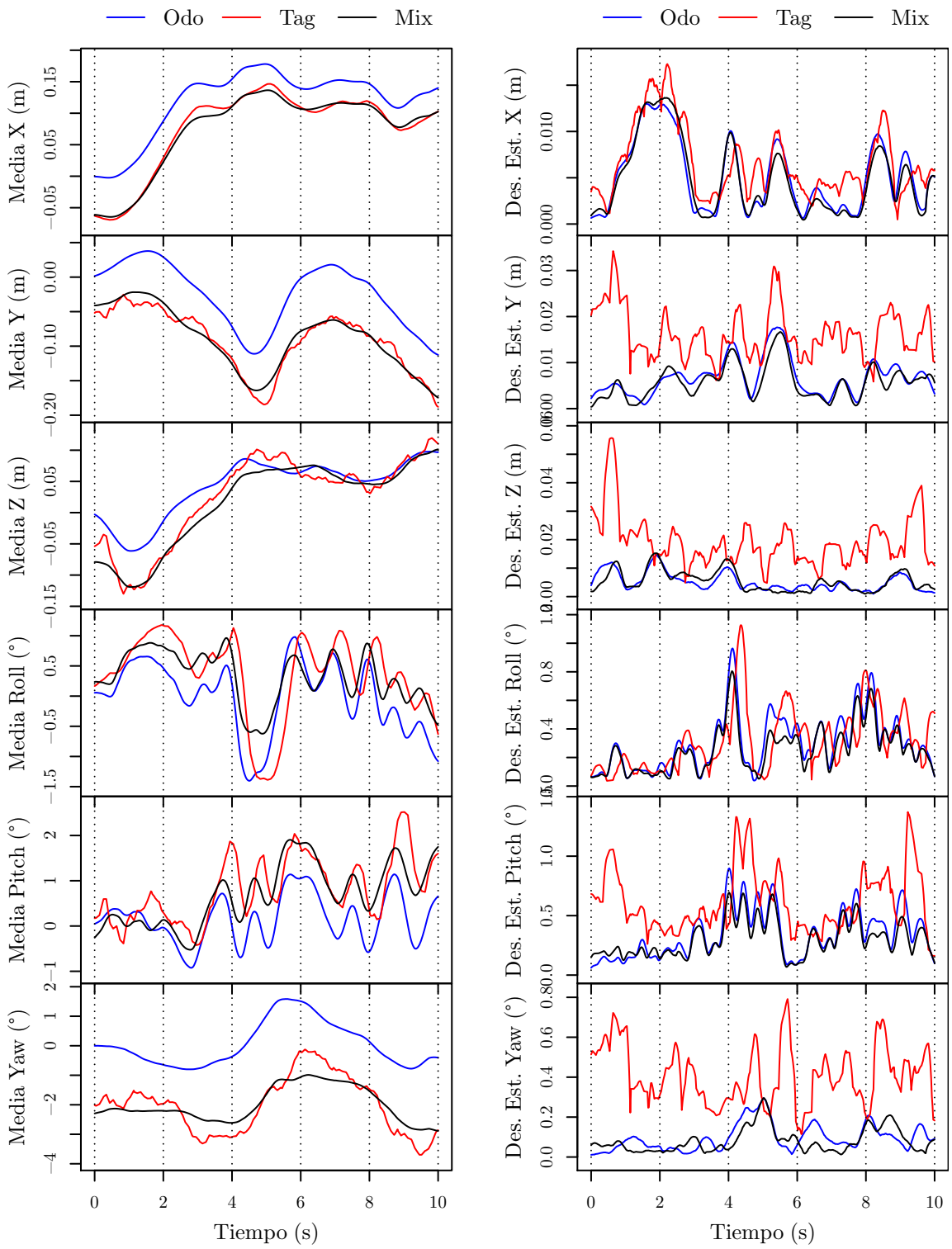


Figura 3.11 Media y desviación estándar de la estimación de la posición y la orientación del AR.Drone 2.0 por medio de diferentes métodos

3.8 Módulo Gráfico

La necesidad de una interfaz gráfica para el sistema se contempló desde sus inicios para tener una mejor visualización de la evolución de las pruebas, aunque su uso realmente no es obligatorio. Se encuentra habilitada por defecto tanto en el modo simulación como en el de implementación, siendo posible desactivarla sólo durante la compilación del código. En caso de ello, toda la información importante pasa a mostrarse en la consola de ejecución.

La librería externa que permitió su creación, se llama OpenGL (*Open Graphics Library*), y es una biblioteca estándar multilenguaje y multiplataforma para programar aplicaciones que produzcan gráficos 2D y 3D. Para que funcione, hace falta integrar la librería al código y también, que el sistema operativo incorpore los archivos de OpenGL vinculados al mismo. Crear una ventana de trabajo en Linux, en Mac OS, o en Windows, difiere porque cada uno habla su propio idioma, y se necesitaría escribir un código específico para cada uno. La ventaja de OpenGL es que ya incorpora la forma de entenderlos, por lo que todo ese proceso es transparente para el desarrollador. Las fuentes [81, 82, 83, 84] resultaron de gran ayuda durante la programación del mismo.

Una captura de pantalla de la interfaz gráfica se puede observar en la Figura 3.12. Esta compuesto por cuatro vistas, una principal que es la tridimensional XYZ, y tres secundarias de los diferentes planos YZ, XZ y XY. La vista principal cuenta además con información acerca de la fecha y hora de la prueba, el estado del sistema -Detenido o Corriendo-, el valor del tiempo de simulación, y los valores de posición y orientación del cuadricóptero. Adicionalmente, se puede desplazar la vista principal, por medio del teclado y el mouse, para obtener una mejor perspectiva.

En relación al escenario, se compone fundamentalmente por cuatro elementos: el dron, la trayectoria deseada, la trayectoria recorrida y la cuadrícula de fondo que sirve de referencia. Todo lo dibujado se realiza por medio de puntos que se conectan para formar rectas, que a su vez crean polígonos y éstos finalmente solidos. Todas las formas curvas son en realidad aproximaciones infinitesimales de las anteriores. Existen formas predefinidas que agilizan el trabajo de construcción y fueron de gran ayuda en la representación del dron. Al tener la forma deseada, se le indica su posición, orientación y tamaño por medio de transformaciones geométricas incorporadas en OpenGL.

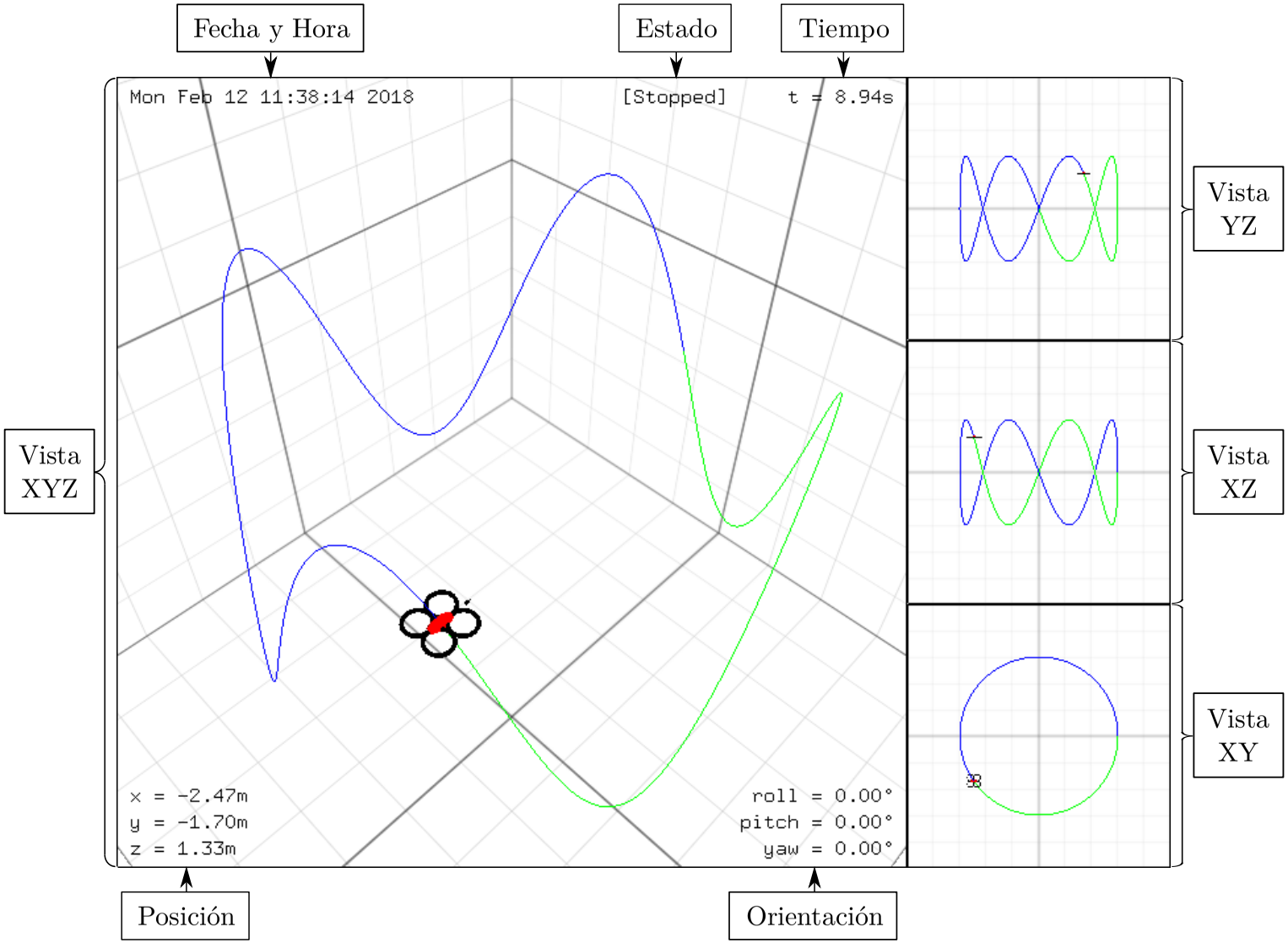


Figura 3.12 Interfaz gráfica del sistema

3.9 Módulo de Entrada y Salida

Finalmente falta explicar qué requiere el sistema como entrada, y qué arroja luego de salida. Cuando el sistema arranca, necesita cargar varios archivos de configuración que incluyen toda la información descrita en las secciones anteriores. El conjunto incluye variables de las pruebas, como el tipo y características de la trayectoria, los parámetros del modelo y el controlador, los tiempos de simulación, y los asociados a la comunicación y el filtro, e incorpora también datos de la aplicación como el tamaño de la ventana de trabajo, las dimensiones del espacio tridimensional y la perspectiva de la vista principal.

Una vez concluida la prueba, el sistema espera que se cierre la aplicación para así generar un archivo de salida con los datos obtenidos en formato de texto simple. Las variables que se guardan son la posición y orientación, y las velocidades y aceleraciones lineales y angulares del cuadricóptero para cada instante de tiempo. La estructura de los datos en el archivo es en forma de matriz, donde las columnas son las variables y las filas los datos que contienen. La separación entre columnas se realiza por un carácter de espacio, y entre las filas por un salto de linea. Esta configuración es de las más sencillas para un posterior manejo con entornos de análisis de datos como R y MatLab.

3.10 Medición del Rendimiento

A fin de comparar el comportamiento del sistema y de los controladores bajo diferentes condiciones, se decidió emplear dos índices de rendimiento basados en el error ampliamente conocidos en el área de control, el IAE (*Integral Absolute Error*) que integra el error absoluto a lo largo del tiempo, y el ISE (*Integral Squared Error*) que integra el cuadrado del error en función del tiempo (3.25). Existen otros índices similares como el ITAE (*Integral Time-weighted Absolute Error*) o el ITSE (*Integral Time-weighted Squared Error*) que no se incluyeron debido a que poseen una fuerte dependencia del tiempo, y en consecuencia penalizan de manera diferente un error al final de la trayectoria que al inicio. Nótese que como regla general, un valor de índice más bajo implica un mejor rendimiento.

$$IAE = \int |e(t)|\, dt \qquad ISE = \int e^2(t)\, dt \tag{3.25}$$

Las expresiones anteriores muestran que el índice IAE afecta por igual los diferentes errores, mientras que el índice ISE penaliza en mayor medida los errores grandes y en menor medida los pequeños. Individualmente, el IAE tiende a producir respuestas mas lentas pero con menos oscilaciones, y el ISE conduce a respuestas mas rápidas pero con oscilaciones persistentes a lo largo del tiempo. Dado que cada índice tiene características relevantes, el estudio buscó un equilibrio al tratar de minimizar, en la medida de lo posible, ambos a la vez.

Capítulo 4

Resultados

En el cuarto capítulo se abarcan las pruebas y resultados del estudio realizado bajo las diferentes modalidades. Se organiza en dos grandes secciones, una para las simulaciones y otra para la implementación con el AR.Drone. Cada sección inicia con la descripción del protocolo de pruebas, seguido de los resultados obtenidos y culmina con las observaciones pertinentes.

4.1 Resultados Simulados

Antes de realizar las pruebas, es necesario definir bajo que condiciones se van a guiar. A continuación se señalan las de mayor importancia:

- El modelo del cuadricóptero se estudia bajo distintas situaciones respecto a las perturbaciones: sin perturbación alguna, con viento a una velocidad constante de $w = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}^T$ m/s, y con ruido gaussiano que tiene media nula y desviación $\sigma_r = 0.1$ m y $\sigma_\eta = 1°$. Las cifras son al menos diez veces mayores que las vistas en la literatura [16, 30].

- Los controladores a evaluar son el Difuso, como controlador inteligente, y el PID, como controlador convencional, aplicados sobre el control de posición y el de orientación bajo las cuatro posibles combinaciones: PID-PID, PID-Difuso,

Difuso-PID y Difuso-Difuso. Los parámetros de los mismos se pueden observar en la Tabla 4.1, obtenidos por experticia en pruebas de ajuste.

- Las trayectorias de interés son el círculo (A.7), la lemniscata (A.9) y el cuadrado (polígono con $n = 4$) (A.13), las cuales reflejan diferentes casos del comportamiento de la pendiente a lo largo de una ruta: continuo-constante, continuo-variable y discontinuo. De las características de configuración se puede señalar que mantienen la posición y orientación de sus primitivas y sólo se escalan por un factor de tres para adecuarlas a las dimensiones del cuadricóptero.

- El estado inicial del cuadricóptero estipula de posición, el punto de comienzo de la trayectoria, y de orientación, los ángulos que le confieren máxima estabilidad. El resto de variables que influyen en cálculos posteriores, como las velocidades y las aceleraciones, se asignan en valores nulos.

- El tiempo de ejecución de las pruebas se establece en 15 s, cantidad que permite al cuadricóptero completar la trayectoria de manera satisfactoria y a la vez resaltar lo mayor posible las deficiencias del control. El intervalo de tiempo entre los ciclos del algoritmo Δt se fija en 20 ms para corresponder al mismo de la implementación.

Tabla 4.1 Parámetros de los controladores para la simulación

	PID			Difuso		
	K_p	K_i	K_d	K_p	K_d	K_o
Posición	5.0	0.0	1.0	5.0	1.0	3.0
Orientación	500.0	0.0	25.0	5.0	0.25	150.0

A manera de ejemplo, en las Figuras 4.1, 4.2 y 4.3 se puede observar el funcionamiento del sistema en algunas modalidades. Las únicas pruebas que emplean valores aleatorios, son las que tienen perturbaciones con ruido, en cuyos casos para calcular las medidas de rendimiento, se requieren una cantidad de ejecuciones que sea estadísticamente significativa. Un total de 100 corridas se definió por ser aceptable, mientras que para el resto de casos, una sola era suficiente. El computador destinado para las simulaciones era una portátil, con procesador Intel Core2Duo CPU T7250 a 2.00 GHz, memoria RAM de 4 GB, tarjeta de vídeo Nvidia GeForce 8600M GT con 256 MB de memoria dedicada y sistema operativo Debian GNU/Linux 8 de 64 bit.

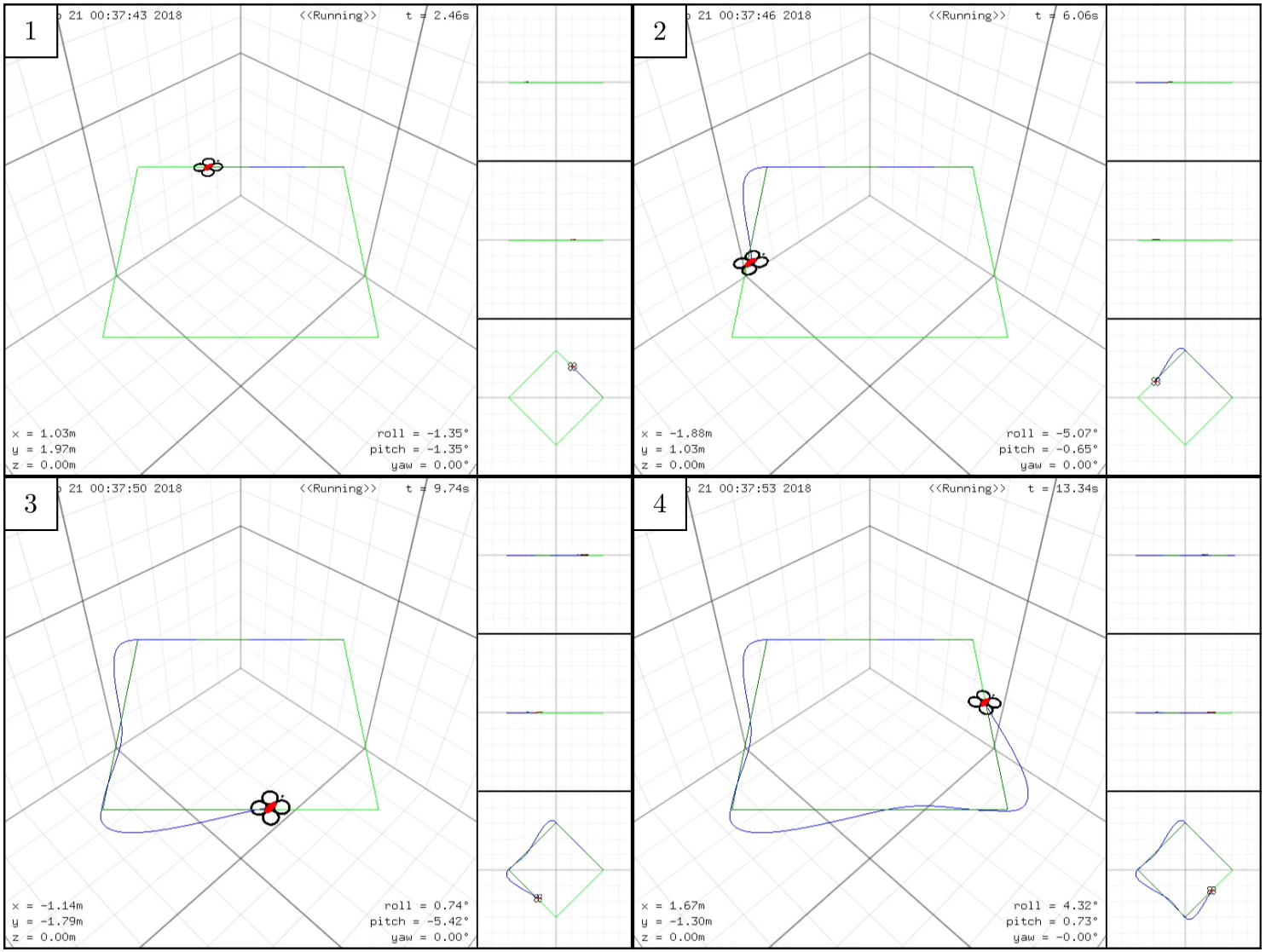


Figura 4.1 Sistema operando en la trayectoria cuadrada y sin perturbaciones

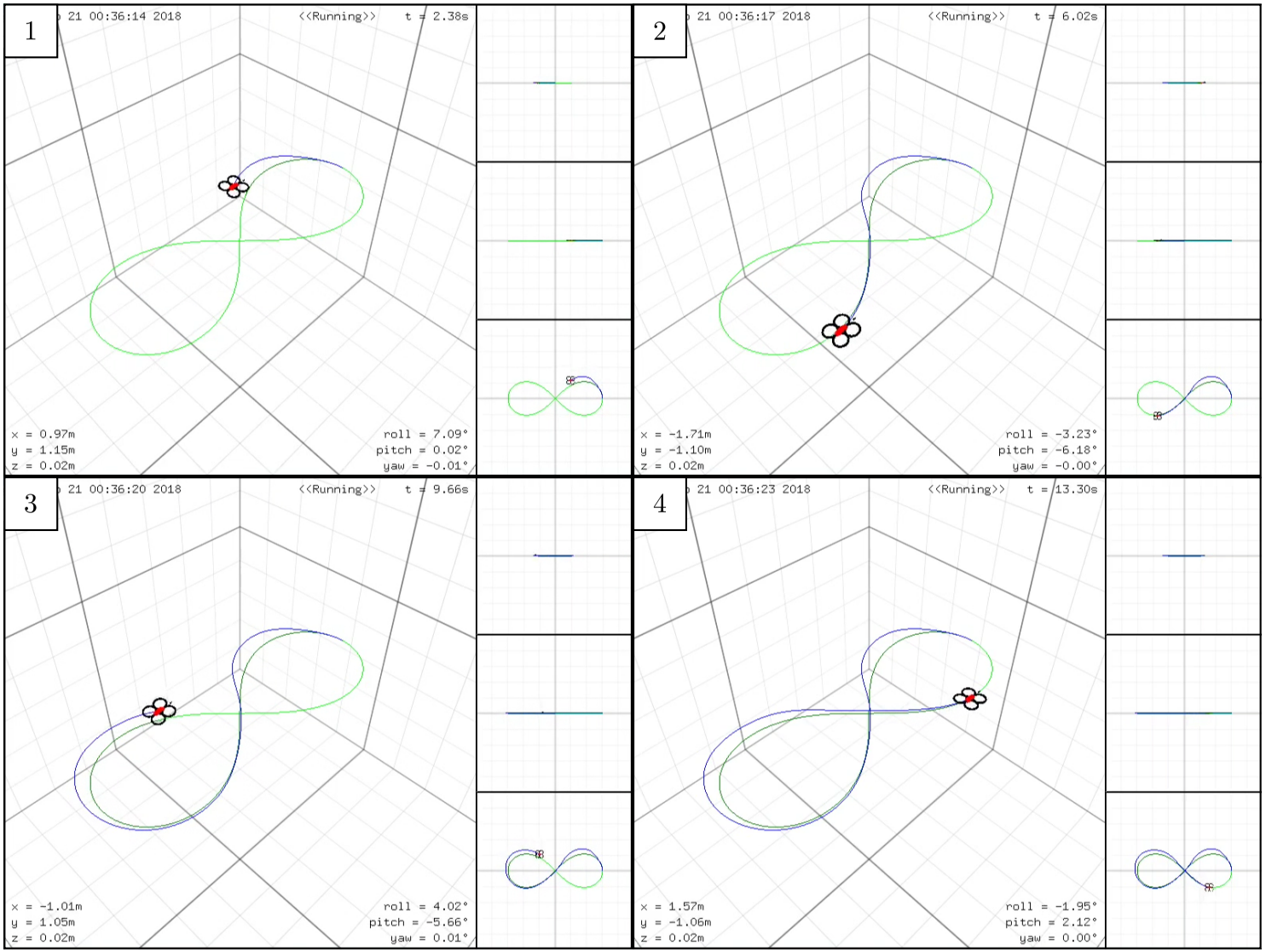


Figura 4.2 Sistema operando en la trayectoria lemniscata y con viento

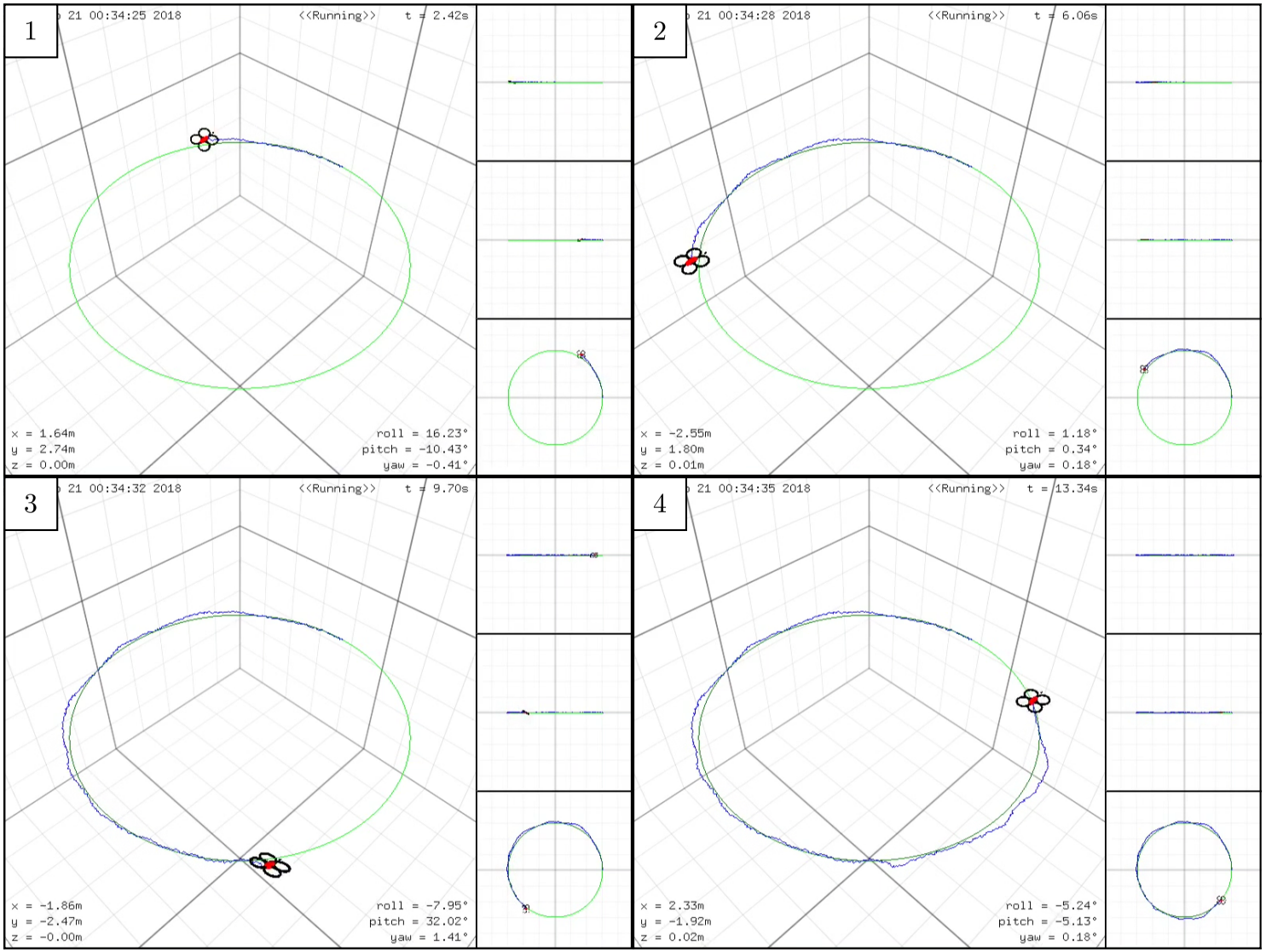


Figura 4.3 Sistema operando en la trayectoria circulo y con ruido

En primera instancia se examinan los resultados de la simulación ideal, es decir, sin perturbación. Los mismos se detallan en la Tabla 4.2 y, para una mejor apreciación, en la Figura 4.4. A simple vista se observa que hay una gran semejanza entre las configuraciones que comparten el control de posición, por lo que la elección del control de orientación no parece relevante. Ambas medidas de rendimiento muestran que los que usan un controlador difuso en el control de posición, tienen mejor desempeño en las trayectorias circulo y lemniscata, mientras que los que emplean un controlador PID, resaltan en la trayectoria cuadrada. El IAE afianza más la diferencia en el circulo y la lemniscata, en cambio el ISE lo hace en la lemniscata y el cuadrado. En un balance general, el rendimiento acumulado es parecido en los diferentes casos.

Luego, como segundo conjunto de resultados, se tienen los que involucran el viento, es decir, una perturbación constante. La Tabla 4.3 precisa los valores obtenidos, y la Figura 4.5 muestra su representación en un gráfico de barras. De manera acorde con los resultados anteriores, hay una notoria similitud entre los casos que incorporan el mismo control de posición, sin que afecte en absoluto el tipo de control de orientación. También es evidente el patrón, donde el controlador difuso resalta en las trayectorias circulo y lemniscata, pero el controlador PID lo hace en la trayectoria cuadrada. Sin embargo, el balance general se inclina a favor del controlador difuso, especialmente porque las diferencias que estaban a favor aumentaron y las que estaban en contra disminuyeron.

Por último, están los resultados relacionados con el ruido, es decir, una perturbación variable. Tal como se observa en la Tabla 4.4, se requiere aplicar los indicadores estadísticos de media y desviación estándar, para analizar los resultados y poder construir la Figura 4.6. De allí se obtiene que la combinación del control de posición y de orientación afectan indiscutiblemente en los resultados. Aquellos donde ambos controles poseen el mismo controlador, mantienen cierta coherencia con la tendencia que había hasta ahora, mientras que los restantes manifiestan un hecho revelador. La configuración Difuso-PID resultó con el peor rendimiento en todos los escenarios, sin embargo, el PID-Difuso mostró un desempeño equiparable al Difuso-Difuso en la trayectoria circulo y lemniscata, y mejor al del PID-PID en la trayectoria cuadrada. Además posee la desviación estándar más pequeña, lo cual le otorga solidez a sus resultados. Por consiguiente, éste fue el de mejor balance general.

Tabla 4.2 Resultados de la simulación ideal (sin perturbación)

	Círculo		Lemniscata		Cuadrado	
	IAE	ISE	IAE	ISE	IAE	ISE
PID-PID	1.936	0.303	2.835	0.753	2.380	0.684
PID-Difuso	1.936	0.304	2.830	0.751	2.378	0.686
Difuso-PID	1.381	0.198	1.973	0.422	2.794	1.062
Difuso-Difuso	1.382	0.198	1.969	0.420	2.782	1.055

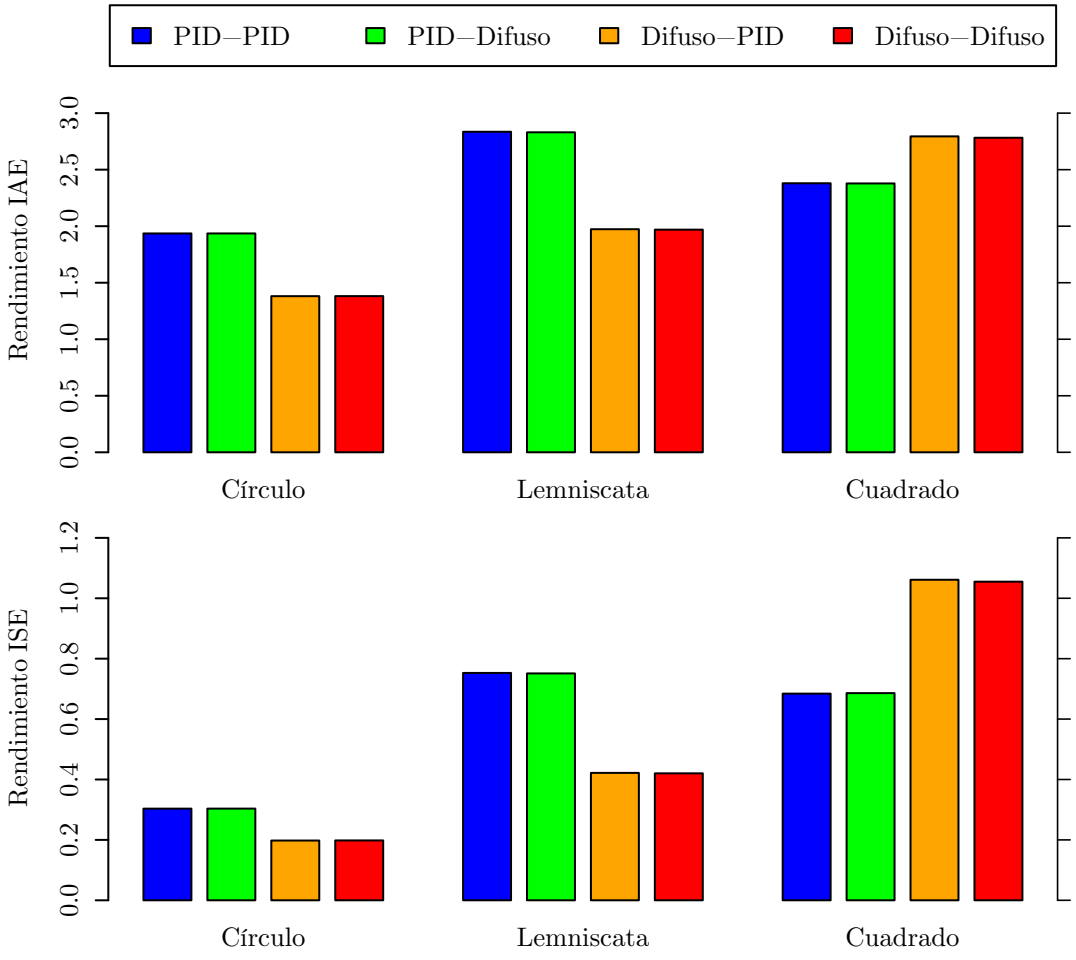


Figura 4.4 Resultados de la simulación ideal (sin perturbación)

Tabla 4.3 Resultados de la simulación con viento (perturbación constante)

	Círculo		Lemniscata		Cuadrado	
	IAE	ISE	IAE	ISE	IAE	ISE
PID-PID	2.526	0.514	3.191	0.909	2.931	0.788
PID-Difuso	2.525	0.514	3.188	0.907	2.930	0.789
Difuso-PID	1.725	0.269	2.389	0.552	3.086	1.071
Difuso-Difuso	1.725	0.269	2.385	0.550	3.080	1.069

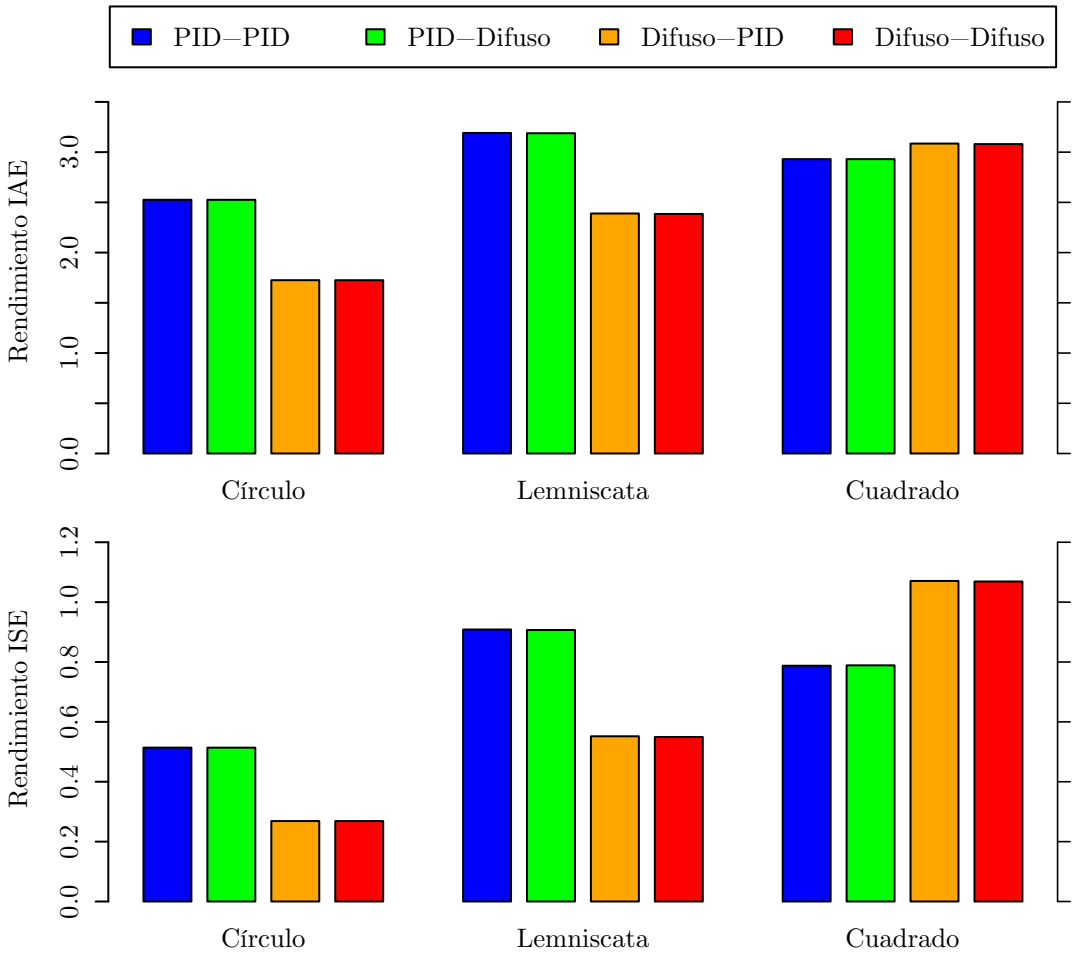


Figura 4.5 Resultados de la simulación con viento (perturbación constante)

Tabla 4.4 Resultados de la simulación con ruido (perturbación variable)

		Círculo		Lemniscata		Cuadrado	
		IAE	ISE	IAE	ISE	IAE	ISE
PID-PID	Media	2.492	0.525	3.284	0.944	3.053	0.918
	Des. Est.	0.289	0.117	0.255	0.149	0.289	0.173
PID-Difuso	Media	2.030	0.339	2.945	0.785	2.484	0.687
	Des. Est.	0.110	0.035	0.142	0.067	0.128	0.062
Difuso-PID	Media	2.617	0.605	3.418	1.081	4.069	1.828
	Des. Est.	0.323	0.149	0.506	0.394	0.650	0.636
Difuso-Difuso	Media	1.755	0.304	2.696	0.711	3.457	1.439
	Des. Est.	0.145	0.053	0.241	0.146	0.404	0.349

En general, los resultados simulados han mostrado que ambos controladores cuentan con una muy buena eficiencia para cumplir la tarea de ejecutar trayectorias. Las pruebas revelaron que el controlador convencional tiene un mejor desempeño bajo los recorridos que incluyeran cambios abruptos y discontinuos, respecto al controlador inteligente. La razón se debe en principio a que el controlador difuso usa en la entrada y la salida, una conversión al dominio difuso que restringe el rango de operación. Las discontinuidades de las trayectorias producen sobrepicos en las entradas del controlador, que no son debidamente correspondidos con la acción de la salida. El controlador PID no tiene esa limitación porque dispone de un rango completo de operación.

Por otro lado, el controlador propuesto evidenció mejores cualidades que su homólogo convencional, para enfrentar recorridos de tipo continuos, ya sea que se mantengan constantes o que varíen en el tiempo. Este tipo de trayectorias comprenden los rangos de operación normales para un dron, especialmente si están involucradas velocidades altas. Bajo este escenario, el controlador difuso es capaz de evidenciar su gran potencial. El éxito en las pruebas va ligado a su diseño que incorpora conocimiento experto del problema, y le confiere características inteligentes que van más allá del control automático del controlador PID. También existen configuraciones que son capaces de combinar las mejores características de ambos controladores, produciendo resultados extraordinarios.

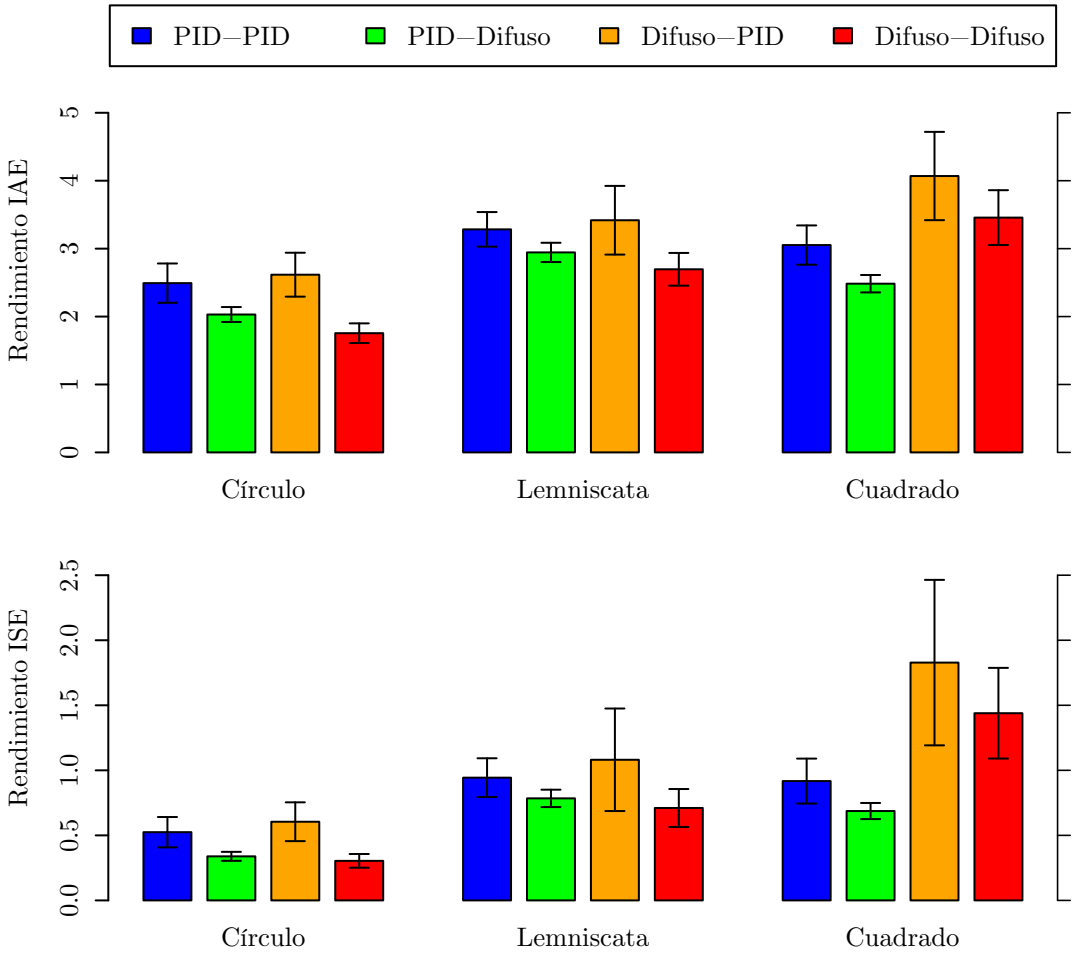


Figura 4.6 Resultados de la simulación con ruido (perturbación variable)

4.2 Resultados Reales

Las condiciones sobre las que se guiaron estas pruebas son similares a las de la sección anterior, sin embargo existen algunas diferencias que se explican a continuación:

- No se establecen tipos de perturbaciones ya que de por sí son inherentes al AR.Drone y al escenario de pruebas, un espacio cerrado de aproximadamente $3 \times 3 \times 2$ m dentro del laboratorio de investigación. Cabe destacar que en el plano frontal del espacio, se ubica la marca o *tag* que sirve de referencia externa, la cual tiene dimensiones 60×60 cm y cuyo centro está alineado con el plano de las trayectorias. La marca empleada es la primera que se observa en la Figura 3.6.

- Los controladores que se ponen a prueba son el Difuso, como controlador inteligente, y el PID, como controlador convencional, pero ya no bajo las mismas combinaciones. Debido a que el AR.Drone se encarga de gran parte del control de orientación, se optó por aplicar el mismo controlador tanto en la posición, como en la fracción del control de orientación que se posee, quedando por consiguiente, dos configuraciones: PID y Difuso. Para obtener el valor de sus parámetros, mostrados en la Tabla 4.5, se partió de los calculados en la sección de simulación, y se le aplicaron ajustes en función de la experiencia adquirida con las pruebas realizadas.

- Las trayectorias siguen siendo el círculo (A.7), la lemniscata (A.9) y el cuadrado (polígono con $n = 4$) (A.13), para tener una comparativa adecuada. Debido al espacio físico disponible, las figuras se ajustaron dentro de una circunferencia de diámetro 1.5 m y dejar así una banda de seguridad de 0.75 m en cada dirección. La altura respecto al suelo se estableció en 80 cm, distancia aproximada en donde el piloto automático del AR.Drone lo estabiliza tras el despegue. Además de las trayectorias establecidas, se agrega un caso especial denominado punto, que tiene como objetivo mostrar el desempeño de los controladores para mantenerse en una posición fija en el espacio, en este caso, el origen del sistema de coordenadas.

- El estado inicial del AR.Drone viene determinado por la posición y orientación que exhiba el cuadricóptero luego del despegue. Desafortunadamente no es un estado que se pueda fijar con exactitud, ya que depende mucho de las perturbaciones del momento y de imperfectos en el proceso de despegue. Aún así, se trata de minimizar estos efectos al posicionarlo justo debajo de donde comienza la trayectoria y de recalibrar la estimación de rotación en el AR.Drone antes de cada prueba. Esto último se hace con la llamada al servicio *ardrone/flattrim* del paquete *ardrone_autonomy*.

- El tiempo de ejecución de las pruebas se establece en 10 s, un poco mayor al que correspondería por la reducción del tamaño de las trayectorias, pero es necesario emplear velocidades no tan elevadas para asegurar la integridad de la plataforma. Respecto al intervalo de tiempo Δt, la documentación recomienda 20 ms para un buen control y pruebas preliminares lo confirmaron. Dicho valor es óptimo ya que valores menores entran en conflicto con el tiempo máximo de ejecución de la aplicación, y valores mayores no efectúan el control lo suficientemente rápido para sacar su máximo potencial.

Tabla 4.5 Parámetros de los controladores para la implementación

	PID			Difuso		
	K_p	K_i	K_d	K_p	K_d	K_o
Posición	1.0	0.0	1.0	1.0	1.0	2.0
Orientación	250.0	0.0	25.0	2.5	0.25	150.0

En las Figuras 4.7, 4.8 y 4.9 se puede observar ejemplos del sistema funcionando. Con relación a las corridas, debe aclararse que se requiere mucho tiempo y esfuerzo para realizar una cantidad estadísticamente significativa en cada prueba. Un solo intento puede consumir entre 10 y 20% de la batería del AR.Drone, y cuando se agota necesita más de 90 min para recargarse. Además, algunas pruebas previas demostraron que niveles de energía inferiores a 40% arrojan resultados poco confiables. Por las razones expuestas se decidió realizar tres intentos de cada prueba y seleccionar sólo la del mejor desempeño mostrado. El computador destinado para la implementación era un equipo de sobremesa, con procesador Intel Core i5 CPU 4570R a 2.7 GHz, memoria RAM de 8 GB, vídeo integrado Intel Haswell y sistema operativo Ubuntu 14.04 LTS de 64 bit.

Una vez hechas las pruebas de implementación, los resultados obtenidos son tabulados y representados gráficamente por medio de la Tabla 4.6 y la Figura 4.10. En consecuencia se observa que bajo el caso especial del punto, ambos controladores se mantuvieron en niveles cercanos de desempeño, siendo mejor por una pequeña fracción el controlador PID. Por el lado de las trayectorias, se tiene que el controlador difuso mostró un mejor rendimiento en las trayectorias círculo y lemniscata, aunque en la trayectoria cuadrada fue superado por el controlador PID. Al realizar un balance general hay una leve ventaja por parte del controlador difuso.

De manera general, tanto el controlador propuesto como el controlador convencional, fueron capaces de enfrentar con éxito los escenarios planteados, lo cual es un indicativo de las capacidades del sistema. En las pruebas implementadas, el patrón recurrente visto en las simulación vuelve hacer aparición. La especialidad del controlador difuso son las trayectorias de tipo continuas, mientras que el controlador PID tiene su fuerte con las que incluyen cambios exagerados, lo cual confirma el análisis efectuado en sección anterior. El controlador propuesto demuestra que en la mayoría de escenarios, es igual o superior en rendimiento que el controlador convencional.

Figura 4.7 Sistema operando en la trayectoria circulo con el AR.Drone

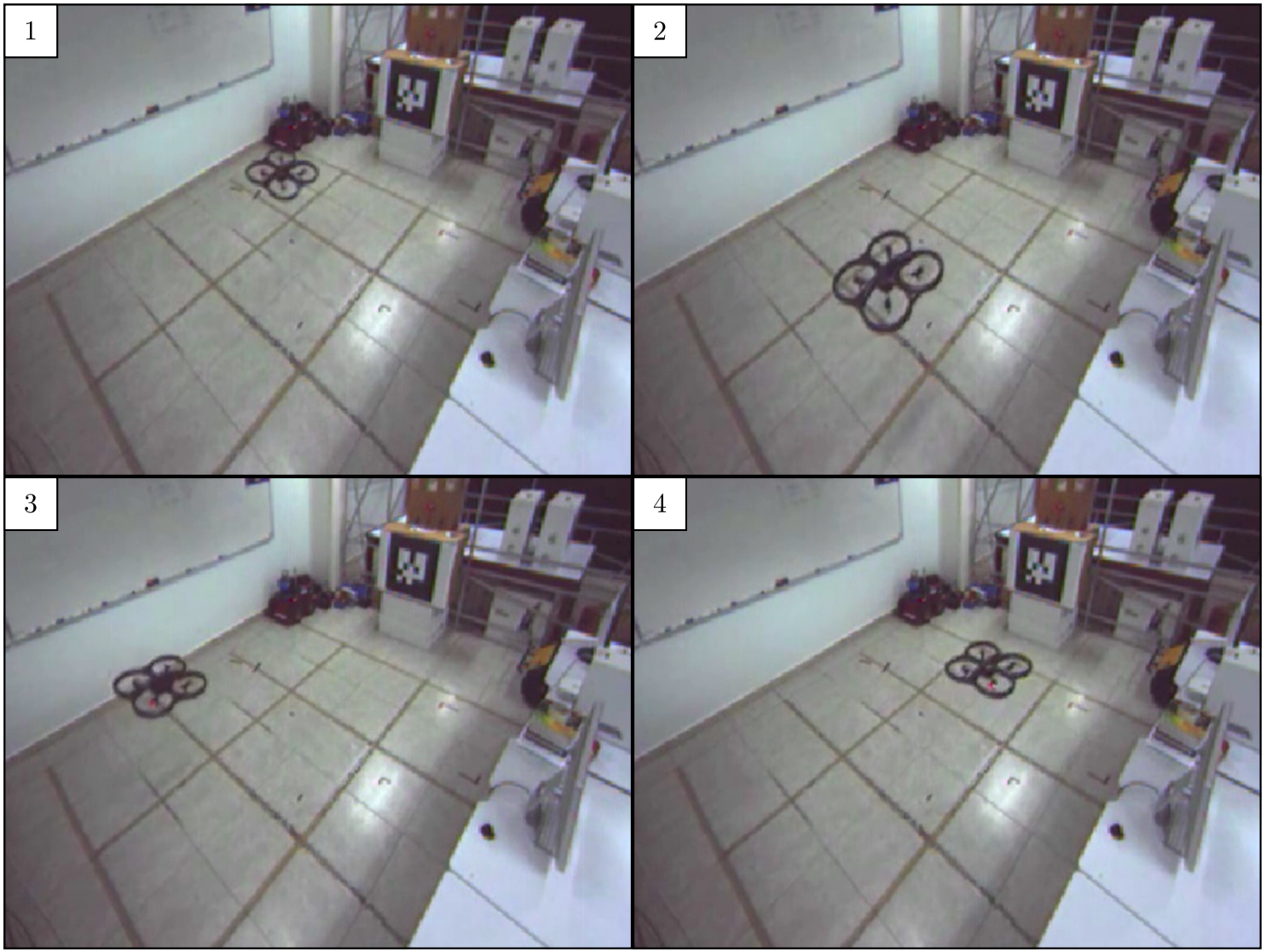


Figura 4.8 Sistema operando en la trayectoria lemniscata con el AR.Drone

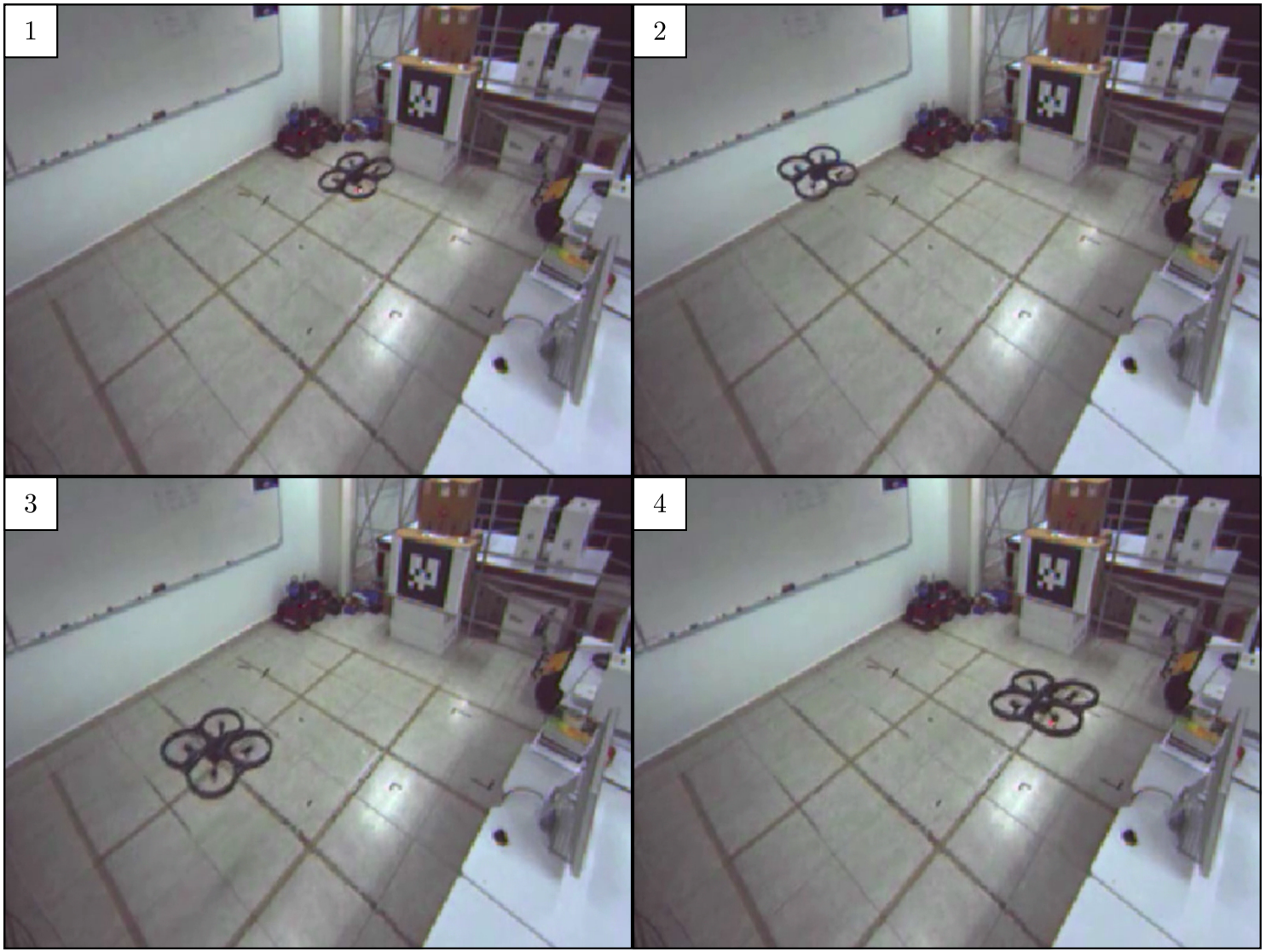


Figura 4.9 Sistema operando en la trayectoria cuadrada con el AR.Drone

Tabla 4.6 Resultados de la implementación real

	Punto		Círculo		Lemniscata		Cuadrado	
	IAE	ISE	IAE	ISE	IAE	ISE	IAE	ISE
PID	0.900	0.096	1.884	0.447	1.739	0.381	1.751	0.364
Difuso	1.009	0.111	1.401	0.243	1.506	0.309	1.864	0.455

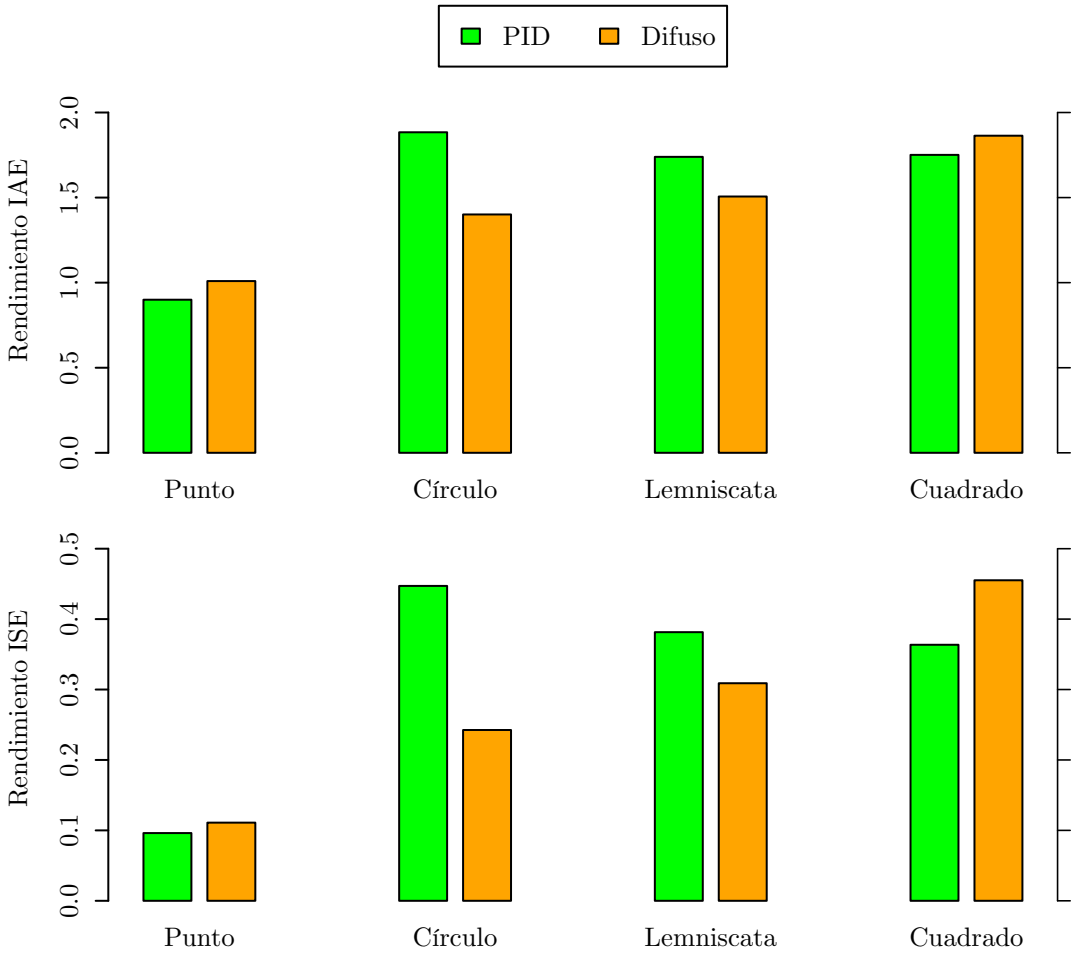


Figura 4.10 Resultados de la implementación real

Capítulo 5

Conclusiones

En el quinto y último capítulo se presentan las conclusiones del presente proyecto. En primer lugar se sintetizan las metas alcanzadas durante el desarrollo de toda la investigación, dejando para el final, algunas perspectivas para una futura continuación del estudio.

5.1 Metas logradas

El estudio de técnicas enfocadas a la ejecución y seguimiento de trayectorias por parte de drones aéreos como los cuadricópteros, es un área de gran oportunidad para la investigación científica. El campo de aplicación es tan amplio y diverso, que aún hoy en día, con todos los trabajos que existen al respecto, sigue latente la necesidad de estudiar nuevas y mejores formas para cumplir la tarea.

El objetivo principal de la presente investigación era el desarrollo de un sistema de control automático capaz de hacer frente a este tipo de aplicación. Asimismo, dos enfoques de control servirían de comparativa para el sistema, uno inteligente por medio de un controlador difuso y otro convencional a través de un controlador PID. Ambas alternativas debían ponerse a prueba sobre diferentes condiciones para establecer un estudio lo suficientemente completo, no solo de manera simulada, sino también en un escenario real con una plataforma robótica.

En un principio para poner en contexto el estudio, se realizó una investigación de aspectos clave como definiciones y características relacionadas, aunado a una revisión bibliográfica de trabajos afines del estado del arte. Este proceso permitió fijar un punto de partida de lo que estaba por venir. La deducción del modelo matemático que rige el comportamiento complejo e inestable de los cuadricópteros, sentó las bases para establecer posteriormente, una metodología para su control. Asimismo, la exploración del AR.Drone y todo lo concerniente a él, permitió adquirir las herramientas para su integración y control en el sistema.

Después, con el conocimiento adquirido se procedió al paso más extenso, el desarrollo del sistema. Durante el proceso, fue necesario establecer una estructura basada en módulos que abarcaran las distintas áreas del problema y así volverlo manejable. Una explicación detallada de todas sus funciones, y las consideraciones tomadas, facilitaron una mejor comprensión de cada uno. En los módulos de control y de estimación, se hicieron aportes novedosos por la metodología aplicada, concretamente en la definición de las saturaciones para generar las señales de control, en la reducción de parámetros de ajuste inherentes al controlador difuso y en la obtención de una buena estimación a partir de la combinación de otras dos con características deficientes.

Posteriormente se establecieron las condiciones que pondrían en evidencia la eficiencia del sistema y en especial, de los controladores del estudio. Las pruebas incluían diferentes trayectorias, varias perturbaciones y distintas combinaciones de los controladores, bajo condiciones extremas para resaltar sus deficiencias. En relación a los parámetros de los controladores, fue necesario realizar un ajuste empírico dirigido a minimizar los índices de rendimiento IAE e ISE de los escenarios planteados. De allí se destaca el papel fundamental que juegan las componentes proporcional y derivativa de ambos controladores para un control adecuado bajo estas condiciones.

Al final, los resultados obtenidos de la simulación y la implementación, evidenciaron que no existía controlador que sobresaliera en todos los casos de estudio, sin embargo, bajo trayectorias con pendiente continua, el difuso mostraba un desempeño mejor respecto al PID. Un hecho que resaltó, fue la posibilidad de generar nuevos y mejores sistemas de control combinando diferentes clases de controladores, como ocurrió con el PID-Difuso que destacó sobre el resto en las pruebas de perturbaciones con ruido. Y de la prueba en suspensión con el AR.Drone, se pudo constatar que un controlador ajustado para seguimiento no necesariamente tiene buen desempeño en regulación.

Como conclusión general, se tiene que el sistema de control desarrollado, es capaz de ejecutar trayectorias definidas de manera exitosa, al menos con los controladores incorporados, bajo diversos entornos simulados y sobre una plataforma real. También se demostró que con un buen ajuste de sus parámetros, el controlador inteligente es capaz de obtener un rendimiento similar e incluso superior al controlador convencional. Asimismo, se deja de contribución, la metodología aplicada para la solución del problema de estudio. Por último y no menos importante, es la primera investigación en su tipo de la institución, lo que deja un antecedente valioso para la posteridad.

5.2 Perspectivas

El gran alcance que tuvo la investigación no implica que el tema esté completamente abarcado. Hay varios aspectos que por falta de tiempo y esfuerzo no se tocaron en el estudio, y que pueden ser el enfoque de futuros proyectos.

En primer lugar, existen otros tipos de controles inteligentes, como los neuronales, que sería interesante desarrollar para establecer nuevas alternativas de investigación. Asimismo, otras clases de control convencional convendría para tener diferentes opciones de comparación. Respecto a las pruebas, se puede acotar la incorporación de un abanico mayor de trayectorias, así como hacer un barrido de diferentes tiempos para completarlas, e incorporar nuevos tipos de perturbaciones. Esto daría una idea más general de las capacidades de los controladores en estudio.

Por otro lado, sería deseable que el proceso de selección de los parámetros de los controladores, se respaldara en la teoría de control o en técnicas de optimización, ya que de esta forma se podrían encontrar los mejores valores para la aplicación. Con respecto al controlador difuso, existen parámetros adicionales, como el número, el tamaño, la forma y la ubicación de las funciones de membresía, que por simplicidad se fijaron en esta investigación y que permitirían otra vía para ajustar su desempeño.

Finalmente, hay modificaciones que pueden aplicarse a las pruebas reales. La incorporación de varias marcas o *tags* alrededor del espacio permitirían ubicar el AR.Drone sin importar su orientación. También el uso de otras plataformas robóticas es un posibilidad latente, debido a la gran compatibilidad que ROS le otorga al sistema.

Apéndice

Este capítulo adicional establece la base completa de trayectorias en el espacio que incluye el sistema desarrollado. Un total de veinte trayectorias diferentes, definidas por funciones paramétricas que tienen como entrada la variable tiempo $t \in [0, 1]$ y como salida la variable posición $x, y, z \in [-1, 1]$, y con una representación tridimensional mostrada en las Figuras A.1 y A.2. Cabe destacar que el presente estudio sólo emplea un subconjunto de estas trayectorias, siendo el excedente un aporte que se deja para futuras investigaciones que las requieran.

- Linea

$$\begin{aligned} x(t) &= -2t + 1 \\ y(t) &= 0 \\ z(t) &= 0 \end{aligned} \tag{A.1}$$

- Parábola

$$\begin{aligned} x(t) &= -2t + 1 \\ y(t) &= 0 \\ z(t) &= 8t^2 - 8t + 1 \end{aligned} \tag{A.2}$$

- Onda de n picos

$$\begin{aligned} x(t) &= -2t + 1 \\ y(t) &= 0 \\ z(t) &= sin(2\pi nt) \end{aligned} \tag{A.3}$$

- Zigzag de n picos

$$\begin{aligned} x(t) &= -2t + 1 \\ y(t) &= 0 \\ z(t) &= 2\left|2\left(nt + \frac{1}{4}\right) - 2\left\lfloor nt + \frac{3}{4}\right\rfloor\right| - 1 \end{aligned} \tag{A.4}$$

- Espiral de n vueltas

$$\begin{aligned} x(t) &= (-t + 1)\cos(2\pi nt) \\ y(t) &= (-t + 1)\sin(2\pi nt) \\ z(t) &= 0 \end{aligned} \tag{A.5}$$

- Limaçon

$$\begin{aligned} x(t) &= \frac{1}{2}\cos(2\pi t) + \frac{1}{2}\cos(4\pi t) \\ y(t) &= \frac{1}{2}\sin(2\pi t) + \frac{1}{2}\sin(4\pi t) \\ z(t) &= 0 \end{aligned} \tag{A.6}$$

- Círculo

$$\begin{aligned} x(t) &= \cos(2\pi t) \\ y(t) &= \sin(2\pi t) \\ z(t) &= 0 \end{aligned} \tag{A.7}$$

- Cacahuete

$$\begin{aligned} x(t) &= \sqrt{1 - \frac{3}{4}\sin^2(2\pi t)}\cos(2\pi t) \\ y(t) &= \sqrt{1 - \frac{3}{4}\sin^2(2\pi t)}\sin(2\pi t) \\ z(t) &= 0 \end{aligned} \tag{A.8}$$

- Lemniscata

$$\begin{aligned} x(t) &= \frac{\cos(2\pi t)}{1+\sin^2(2\pi t)} \\ y(t) &= \frac{\cos(2\pi t)\sin(2\pi t)}{1+\sin^2(2\pi t)} \\ z(t) &= 0 \end{aligned} \tag{A.9}$$

- Pescado

$$\begin{aligned} x(t) &= \frac{19}{20}\cos(2\pi t) - \frac{19}{20\sqrt{2}}\sin^2(2\pi t) + \frac{1}{20} \\ y(t) &= \frac{19}{20}\cos(2\pi t)\sin(2\pi t) \\ z(t) &= 0 \end{aligned} \tag{A.10}$$

- Flor de n pétalos

$$\begin{aligned} x(t) &= \cos(n\pi t)\cos(k\pi t) \\ y(t) &= \cos(n\pi t)\sin(k\pi t) \\ z(t) &= 0 \end{aligned} \qquad k = \begin{cases} 1 & \text{si } n \text{ es impar} \\ 2 & \text{si } n \text{ es par} \end{cases} \tag{A.11}$$

- Estrella

$$\begin{aligned} x(t) &= \frac{3}{5}\cos(4\pi t) + \frac{2}{5}\cos(6\pi t) \\ y(t) &= \frac{3}{5}\sin(4\pi t) - \frac{2}{5}\sin(6\pi t) \\ z(t) &= 0 \end{aligned} \tag{A.12}$$

- Polígono de n lados

$$\begin{aligned} x(t) = &-(2nt - 2\lfloor nt \rfloor - 1)\sin\left(\frac{\pi}{n}\right)\sin\left(\frac{\pi}{n}(2\lfloor nt \rfloor + 1)\right) \\ &+ \cos\left(\frac{\pi}{n}\right)\cos\left(\frac{\pi}{n}(2\lfloor nt \rfloor + 1)\right) \\ y(t) = &\,(2nt - 2\lfloor nt \rfloor - 1)\sin\left(\frac{\pi}{n}\right)\cos\left(\frac{\pi}{n}(2\lfloor nt \rfloor + 1)\right) \\ &+ \cos\left(\frac{\pi}{n}\right)\sin\left(\frac{\pi}{n}(2\lfloor nt \rfloor + 1)\right) \\ z(t) = &\,0 \end{aligned} \tag{A.13}$$

- Corazón

$$\begin{aligned} x(t) &= \frac{13}{17}\cos(2\pi t) + \frac{5}{17}\cos(4\pi t) - \frac{2}{17}\cos(6\pi t) + \frac{1}{17}\cos(8\pi t) \\ y(t) &= \frac{16}{17} sin^3(2\pi t) \\ z(t) &= 0 \end{aligned} \tag{A.14}$$

- Hélice de n vueltas

$$\begin{aligned} x(t) &= \cos(2\pi n t) \\ y(t) &= \sin(2\pi n t) \\ z(t) &= -2t + 1 \end{aligned} \tag{A.15}$$

- Remolino de n vueltas

$$\begin{aligned} x(t) &= (-t+1)\cos(2\pi n t) \\ y(t) &= (-t+1)\sin(2\pi n t) \\ z(t) &= -2t + 1 \end{aligned} \tag{A.16}$$

- Toroide de n vueltas

$$\begin{aligned} x(t) &= \left(\frac{3}{4} + \frac{1}{4}\cos(k\pi n t)\right)\cos(2k\pi t) \\ y(t) &= \left(\frac{3}{4} + \frac{1}{4}\cos(k\pi n t)\right)\sin(2k\pi t) \qquad k = \begin{cases} 1 & \text{si } n \text{ es par} \\ 2 & \text{si } n \text{ es impar} \end{cases} \\ z(t) &= \frac{1}{4}\sin(k\pi n t) \end{aligned} \tag{A.17}$$

- Lissajous

$$\begin{aligned} x(t) &= \cos(6\pi t) \\ y(t) &= \sin(4\pi t) \\ z(t) &= \sin(14\pi t) \end{aligned} \tag{A.18}$$

- Anillo de n picos

$$\begin{aligned} x(t) &= \cos(2k\pi t) \\ y(t) &= \sin(2k\pi t) \qquad k = \begin{cases} 1 & \text{si } n \text{ es par} \\ 2 & \text{si } n \text{ es impar} \end{cases} \\ z(t) &= \sin(k\pi n t) \end{aligned} \tag{A.19}$$

- Corona de n picos

$$
\begin{aligned}
x(t) &= -\left(2knt - 2\lfloor knt \rfloor - 1\right)\sin\left(\frac{\pi}{n}\right)\sin\left(\frac{\pi}{n}\left(2\lfloor knt \rfloor + 1\right)\right) \\
&\quad + \cos\left(\frac{\pi}{n}\right)\cos\left(\frac{\pi}{n}\left(2\lfloor knt \rfloor + 1\right)\right) \\
y(t) &= \left(2knt - 2\lfloor knt \rfloor - 1\right)\sin\left(\frac{\pi}{n}\right)\cos\left(\frac{\pi}{n}\left(2\lfloor knt \rfloor + 1\right)\right) \\
&\quad + \cos\left(\frac{\pi}{n}\right)\sin\left(\frac{\pi}{n}\left(2\lfloor knt \rfloor + 1\right)\right) \\
z(t) &= 2\left|knt - 2\left\lfloor \frac{1}{2}knt + \frac{1}{2}\right\rfloor\right| - 1 \qquad\qquad k = \begin{cases} 1 & \text{si } n \text{ es par} \\ 2 & \text{si } n \text{ es impar} \end{cases}
\end{aligned}
\tag{A.20}
$$

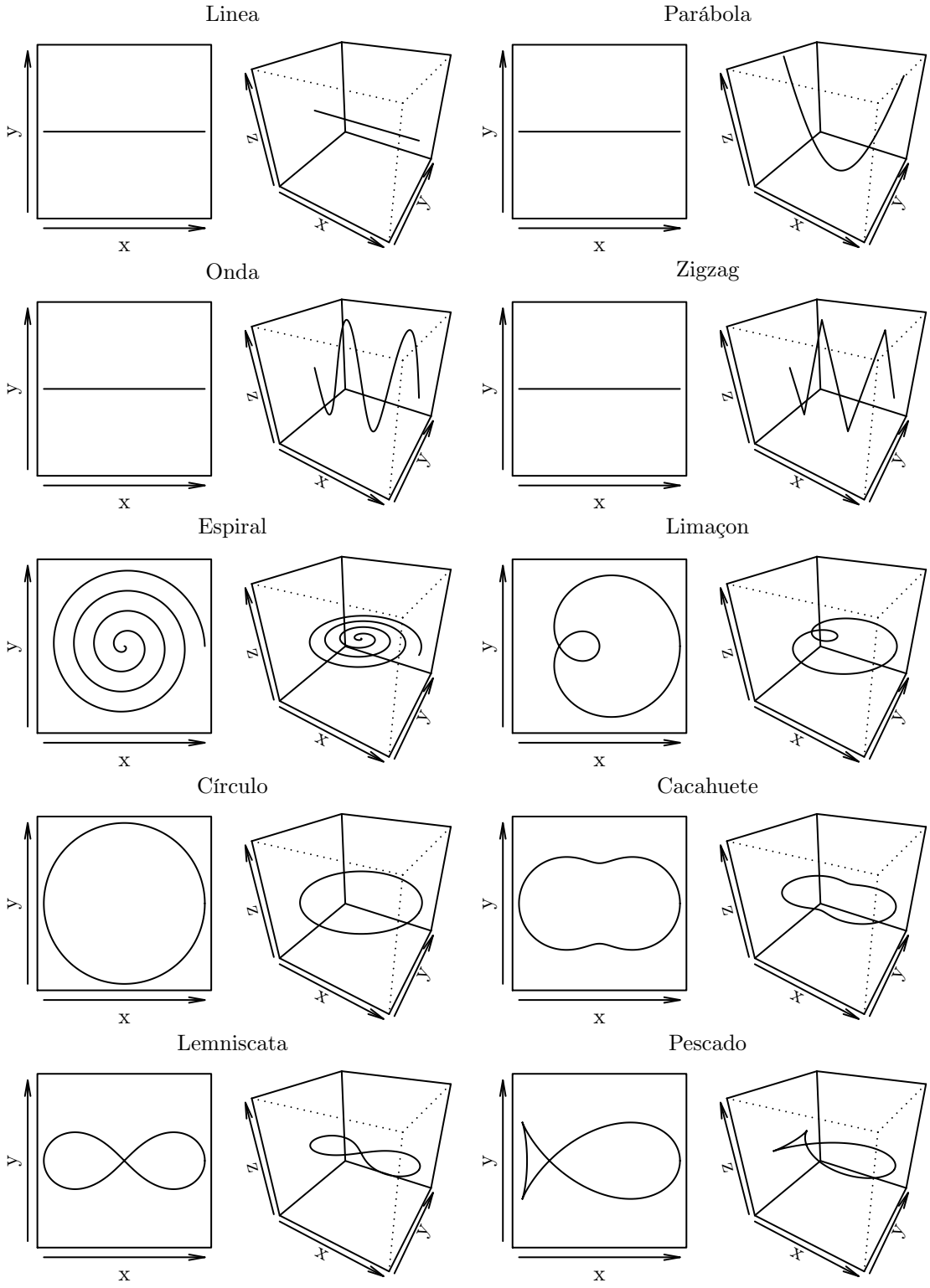


Figura A.1 Conjunto de trayectorias

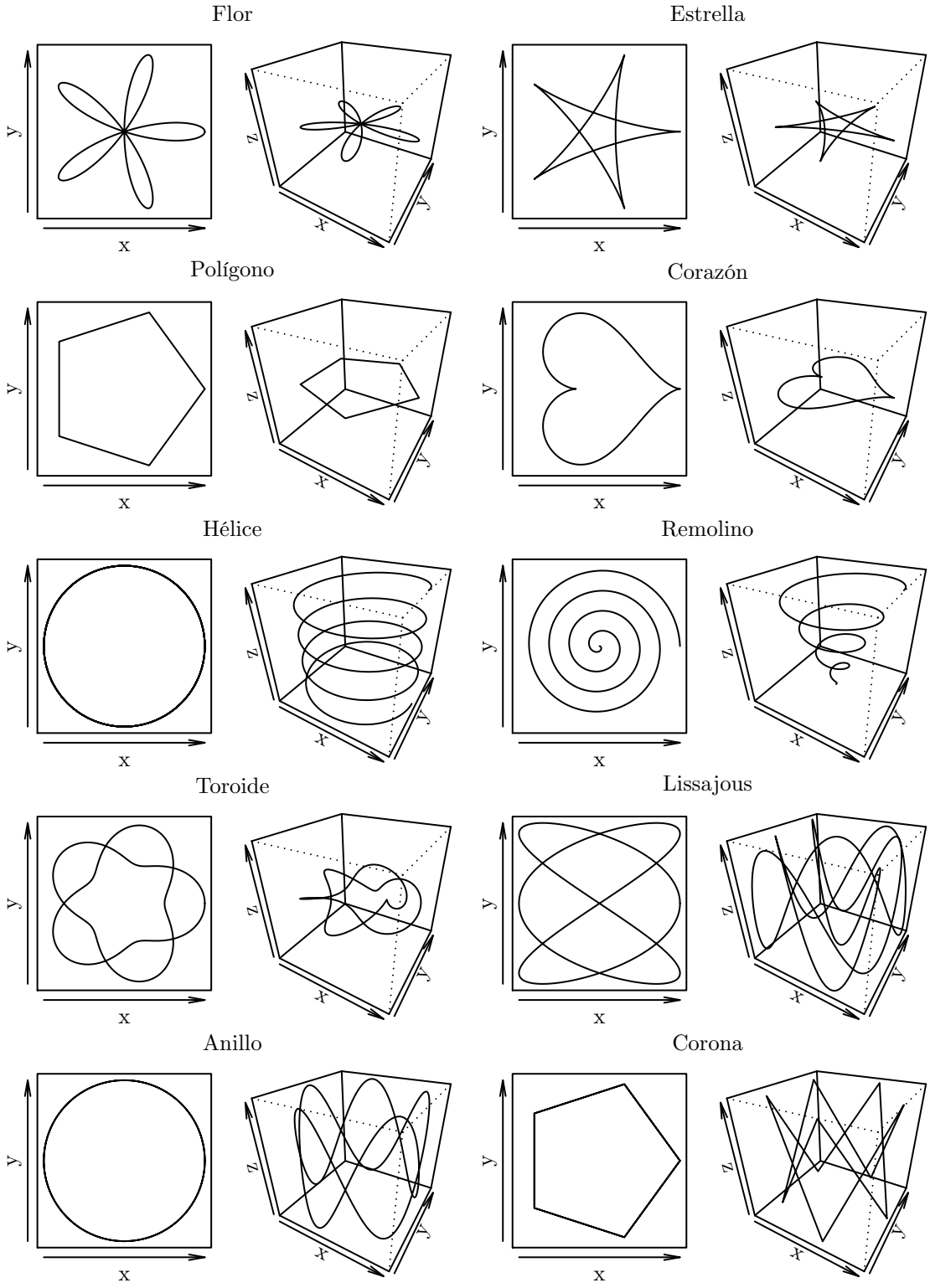


Figura A.2 Conjunto de trayectorias (continuación)

Referencias

[1] J. Santiaguillo-Salinas y E. Aranda-Bricaire, «Seguimiento de trayectorias para un helicóptero de 4 rotores AR.Drone 2.0 utilizando ROS», *Memorias del XVI Congreso Latinoamericano de Control Automático, CLCA 2014*, 2014.

[2] M. J. Reinoso, L. I. Minchala, P. Ortiz, D. F. Astudillo y D. Verdugo, «Trajectory tracking of a quadrotor using sliding mode control», *IEEE Latin America Transactions*, vol. 14, n.º 5, págs. 2157-2166, 2016.

[3] M. W. Mueller y R. D'Andrea, «A model predictive controller for quadrocopter state interception», en *Control Conference (ECC), 2013 European*, IEEE, 2013, págs. 1383-1389.

[4] M. Hehn y R. D'Andrea, «An iterative learning scheme for high performance, periodic quadrocopter trajectories», en *Control Conference (ECC), 2013 European*, IEEE, 2013, págs. 1799-1804.

[5] ——, «A frequency domain iterative learning algorithm for high-performance, periodic quadrocopter maneuvers», *Mechatronics*, vol. 24, n.º 8, págs. 954-965, 2014.

[6] M. A. Rosaldo-Serrano, J. Santiaguillo-Salinas y E. Aranda-Bricaire, «Modelado y control mediante backstepping de un AR.Drone 2.0», en *Memorias del XVIII Congreso Mexicano de Robótica 2016*, AMRob, 2016.

[7] A. Zulu y S. John, «A review of control algorithms for autonomous quadrotors», *arXiv preprint arXiv:1602.02622*, 2016.

[8] R. V. Lopes, P. Santana, G. Borges y J. Ishihara, «Model predictive control applied to tracking and attitude stabilization of a VTOL quadrotor aircraft», en *21st International Congress of Mechanical Engineering*, 2011, págs. 176-185.

[9] G. V. Raffo, M. G. Ortega y F. R. Rubio, «An integral predictive/nonlinear H∞ control structure for a quadrotor helicopter», *Automatica*, vol. 46, n.º 1, págs. 29-39, 2010.

[10] O. Doukhi, A. R. Fayjie y D. J. Lee, «Intelligent controller design for quad-rotor stabilization in presence of parameter variations», *Journal of Advanced Transportation*, vol. 2017, 2017.

[11] M. W. Mueller, M. Hehn y R. D'Andrea, «A computationally efficient motion primitive for quadrocopter trajectory generation», *IEEE Transactions on Robotics*, vol. 31, n.º 6, págs. 1294-1310, 2015.

[12] T. S. Kim, K. Stol y V. Kecman, «Control of 3 DOF quadrotor model», en *Robot Motion and Control 2007*, Springer, 2007, págs. 29-38.

[13] M. W. Mueller, M. Hehn y R. D'Andrea, «A computationally efficient algorithm for state-to-state quadrocopter trajectory generation and feasibility verification», en *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, IEEE, 2013, págs. 3480-3486.

[14] Y. Bouktir, M. Haddad y T. Chettibi, «Trajectory planning for a quadrotor helicopter», en *Control and Automation, 2008 16th Mediterranean Conference on*, IEEE, 2008, págs. 1258-1263.

[15] L. V. Santana, A. S. Brandao, M. Sarcinelli-Filho y R. Carelli, «A trajectory tracking and 3D positioning controller for the AR.Drone quadrotor», en *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, IEEE, 2014, págs. 756-767.

[16] E. Altuğ y B. Erginer, «Design and implementation of a hybrid fuzzy logic controller for a quadrotor VTOL vehicle», *International Journal of Control, Automation and Systems*, vol. 10, n.º 1, págs. 61-70, 2012.

[17] M. Santos, «Un enfoque aplicado del control inteligente», *Revista Iberoamericana de Automática e Informática Industrial RIAI*, vol. 8, n.º 4, págs. 283-296, 2011.

[18] A. Mokhtari y A. Benallegue, «Dynamic feedback controller of euler angles and wind parameters estimation for a quadrotor unmanned aerial vehicle», en *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, IEEE, vol. 3, 2004, págs. 2359-2366.

[19] T. Lee, M. Leoky y N. H. McClamroch, «Geometric tracking control of a quadrotor UAV on SE(3)», en *Decision and Control (CDC), 2010 49th IEEE Conference on*, IEEE, 2010, págs. 5420-5425.

[20] P. Castillo, R. Lozano y A. Dzul, «Stabilization of a mini-rotorcraft having four rotors», en *Conference on Intelligent Robots and Systems, 2004*, IEEE, vol. 3, 2004, págs. 2693-2698.

[21] L. M. Argentim, W. C. Rezende, P. E. Santos y R. A. Aguiar, «PID, LQR and LQR-PID on a quadcopter platform», en *Informatics, Electronics & Vision (ICIEV), 2013 International Conference on*, IEEE, 2013, págs. 1-6.

[22] D. Xu, L. Wang, G. Li y L. Guo, «Modeling and trajectory tracking control of a quad-rotor UAV», en *Proceedings of the 2012 International Conference on Computer Application and System Modeling. Atlantis Press*, 2012.

[23] A. P. Schoellig, C. Wiltsche y R. D'Andrea, «Feed-forward parameter identification for precise periodic quadrocopter motions», en *American Control Conference (ACC), 2012*, IEEE, 2012, págs. 4313-4318.

[24] S. E. Martínez y M. Tomás-Rodríguez, «Seguimiento de trayectorias tridimensionales de un quadrotor mediante control PVA», *Revista Iberoamericana de Automática e Informática Industrial RIAI*, vol. 11, n.º 1, págs. 54-67, 2014.

[25] M. Santos, V. Lopez y F. Morata, «Intelligent fuzzy controller of a quadrotor», en *International Conference on Intelligent Systems and Knowledge Engineering (ISKE), 2010*, IEEE, 2010, págs. 141-146.

[26] M. Santos, «Aplicaciones exitosas de control inteligente a casos reales», *Science Direct, Revista Iberoamericana de Automática e Informática Industrial*, 2011.

[27] S. J. Baek, Y. P. Jeon, U. R. Cho, J. H. Park, D. J. Lee y K. T. Chong, «Intelligent control system design of a unmanned quadrotor robot», en *Applied Mechanics and Materials*, Trans Tech Publ, vol. 548, 2014, págs. 917-921.

[28] R. San Martin, A. Barrientos, P. Gutierrez y J. del Cerro, «Neural networks training architecture for UAV modelling», en *World Automation Congress 2006*, IEEE, 2006, págs. 1-6.

[29] S. Bouabdallah, A. Noth y R. Siegwart, «PID vs LQ control techniques applied to an indoor micro quadrotor», en *Intelligent Robots and Systems (IROS), 2004. RSJ International Conference on*, IEEE, vol. 3, 2004, págs. 2451-2456.

[30] A. C. Satici, H. Poonawala y M. W. Spong, «Robust optimal control of quadrotor UAVs», *IEEE Access*, vol. 1, págs. 79-93, 2013.

[31] C. Caceres, D. Amaya y J. M. Rosário, «Simulation, model and control of a quadcopter AR Drone 2.0», *International Review of Mechanical Engineering (IREME)*, vol. 10, n.º 3, págs. 197-202, 2016.

[32] M. E. Parra Muñoz, E. L. Feitosa Fortaleza y J. M. Alves da Silva, «Modelamiento matemático y control de un helicóptero de cuatro motores», *Scientia et technica*, vol. 18, n.º 4, págs. 672-681, 2013.

[33] W. Aguilar, R. Costa Castelló, C. Angulo Bahón y L. Molina, «Control autónomo de cuadricópteros para seguimiento de trayectorias», en *Revista Digital Congreso de Ciencia y Tecnología: Memorias. Sesiones Técnicas*, 2014, págs. 140-144.

[34] V. Indrawati, A. Prayitno y T. A. Kusuma, «Waypoint navigation of AR.Drone quadrotor using fuzzy logic controller», *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 13, n.º 3, págs. 930-939, 2015.

[35] A. Prayitno, V. Indrawati y G. Utomo, «Trajectory tracking of AR.Drone quadrotor using fuzzy logic controller», *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 12, n.º 4, págs. 819-828, 2014.

[36] I. Sa y P. Corke, «Estimation and control for an open-source quadcopter», en *Proceedings of the Australasian Conference on Robotics and Automation 2011*, 2011.

[37] I. Dryanovski, R. G. Valenti y J. Xiao, «An open-source navigation system for micro aerial vehicles», *Autonomous Robots*, vol. 34, n.º 3, págs. 177-188, 2013.

[38] Í. Viana, L. Santana, D. A. dos Santos y L. Goes, «Experimental validation of a trajectory tracking control using the AR.Drone quadrotor», en *IX Congresso Nacional de Engenharia Mecânica, Fortaleza - Ceará*, ResearchGate, 2016.

[39] P. Vílez, N. Certad y E. Ruiz, «Trajectory generation and tracking using the AR.Drone 2.0 quadcopter UAV», en *Robotics Symposium (LARS) and 2015 3rd Brazilian Symposium on Robotics (LARS-SBR), 2015 12th Latin American*, IEEE, 2015, págs. 73-78.

[40] J. Engel, J. Sturm y D. Cremers, «Scale-aware navigation of a low-cost quadrocopter with a monocular camera», *Robotics and Autonomous Systems*, vol. 62, n.º 11, págs. 1646-1656, 2014.

[41] R. A. Española, *Diccionario de la lengua española*, 23.ª ed. Real Academia Española, 2014, vol. 2.

[42] L. R. García Carrillo, A. E. Dzul López, R. Lozano y C. Pégard, *Quad rotorcraft control: vision-based hovering and navigation*. Springer Science & Business Media, 2012.

[43] G. De Bothezat, *De Bothezat Quadrotor*, http://www.nps.gov/archive/edis/edisonia/graphics/04300002.jpg, 1923.

[44] P. C. Hughes, *Spacecraft attitude dynamics*. Courier Corporation, 2012.

[45] P. H. Zipfel, *Modeling and simulation of aerospace vehicle dynamics*. American Institute of Aeronautics y Astronautics, 2009.

[46] Parrot, *Página oficial AR.Drone 2.0 - AR.Freeflight*, http://global.parrot.com/mx/productos/ardrone-2, Consultado 2017-12-01, 2017.

[47] ——, *Parrot for developers*, http://developer.parrot.com, Consultado 2017-12-01, 2016.

[48] Gauth, *Introduction to the AR.Drone SDK*, http://gauth.fr/2011/09/introduction-to-the-ar-drone-sdk, Consultado 2017-12-01, 2010.

[49] Parrot, *AR Drone 2.0: especificaciones técnicas*, http://ardrone-2.es/especificaciones-ar-drone-2, Consultado 2017-12-01, 2017.

[50] Wikipedia, *Parrot AR.Drone*, https://es.wikipedia.org/wiki/Parrot_AR.Drone, Consultado 2017-12-01, 2017.

[51] Robotshop, *RB-Par-14 Parrot AR.Drone 2.0 quadcopter*, http://www.robotshop.com/media/files/PDF/datasheet-pf721000.pdf, Consultado 2017-12-01, 2012.

[52] PaparazziUAV, *AR Drone 2*, https://wiki.paparazziuav.org/wiki/AR_Drone_2, Consultado 2017-12-01, 2017.

[53] P.-J. Bristeau, F. Callou, D. Vissiere y N. Petit, «The navigation and control technology inside the AR.Drone micro UAV», *IFAC Proceedings Volumes*, vol. 44, n.º 1, págs. 1477-1484, 2011.

[54] Parrot, *AR.Drone developer guide*, English, 2012.

[55] F. Golnaraghi y B. C. Kuo, *Automatic control systems*. John Wiley & Sons, Inc, 2010.

[56] S. Y. Nof, *Springer handbook of automation*. Springer Science & Business Media, 2009.

[57] S. Bennett, «A brief history of automatic control», *IEEE Control Systems*, vol. 16, n.º 3, págs. 17-25, 1996.

[58] L. A. Sánchez Galindo, «Sistemas de control adaptativo utilizando inteligencia computacional», Tesis Profesional, Universidad de Guanajuato, 2016.

[59] F. R. Rubio y M. J. L. Sánchez, *Control adaptativo y robusto*. Universidad de Sevilla, 1996, vol. 9.

[60] E. Villota, *Control Moderno y Óptimo*. Springer, 2013.

[61] R. Galán, A. Jiménez, R. Sanz y F. Matía, «Control inteligente», *Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial*, vol. 4, n.º 10, 2000.

[62] K. Fu, «Learning control systems and intelligent control systems: an intersection of artifical intelligence and automatic control», *IEEE Transactions on Automatic Control*, vol. 16, n.º 1, págs. 70-72, 1971.

[63] K. S. Narendra y S. Mukhopadhyay, «Intelligent control using neural networks», *IEEE Control Systems*, vol. 12, n.º 2, págs. 11-18, 1992.

[64] K. M. Passino, «Intelligent control for autonomous systems», *IEEE spectrum*, vol. 32, n.º 6, págs. 55-62, 1995.

[65] E. Menegatti, N. Michael, K. Berns y H. Yamaguchi, *Intelligent Autonomous Systems 13: Proceedings of the 13th International Conference IAS-13*. Springer, 2015, vol. 302.

[66] N. Minorsky, «Directional stability of automatically steered bodies», *Naval Engineers Journal*, vol. 32, n.º 2, 1922.

[67] K. J. Åström y T. Hägglund, *PID controllers: theory, design, and tuning*. Isa Research Triangle Park, NC, 1995, vol. 2.

[68] L. A. Zadeh, «Fuzzy sets», *Information and control*, vol. 8, n.º 3, págs. 338-353, 1965.

[69] E. H. Mamdani y S. Assilian, «An experiment in linguistic synthesis with a fuzzy logic controller», *International journal of man-machine studies*, vol. 7, n.º 1, págs. 1-13, 1975.

[70] P. Dadios Elmer, *Fuzzy logic–controls, concepts, theories and applications*. InTech, 2012.

[71] C. Riccio, *glm Manual*. G-Truc Creation, 2016.

[72] I. González, S. Salazar, J. Torres, R. Lozano y H. Romero, «Real-time attitude stabilization of a mini-uav quad-rotor using motor speed feedback», *Journal of Intelligent & Robotic Systems*, vol. 70, n.º 1-4, págs. 93-106, 2013.

[73] Parrot, *Parrot AR.Drone2.0 manual de usuario*, Español, 2012.

[74] S. Mađdarić, *C++ Fuzzy Logic API + simple DSL*, https://www.codeproject.com/Articles/316668/Cplusplus-Fuzzy-Logic-API-plus-Simple-DSL, Consultado 2017-12-01, 2012.

[75] M. Monajjemi, *ardrone_autonomy*, http://wiki.ros.org/ardrone_autonomy, Consultado 2017-12-01, 2015.

[76] ——, *ardrone_autonomy*, https://github.com/AutonomyLab/ardrone_autonomy, Consultado 2017-12-01, 2016.

[77] ——, *ardrone_autonomy documentation*, https://ardrone-autonomy.readthedocs.io, Consultado 2017-12-01, 2014.

[78] S. Niekum, *ar_track_alvar*, http://wiki.ros.org/ar_track_alvar, Consultado 2017-12-01, 2016.

[79] ——, *ar_track_alvar*, https://github.com/ros-perception/ar_track_alvar, Consultado 2017-12-01, 2017.

[80] S. K. Mitra e Y. Kuo, *Digital signal processing: a computer-based approach*. McGraw-Hill Higher Education, 2006, vol. 2.

[81] O. García y A. Guevara, *Introducción a la programación gráfica con OpenGL*, 2004.

[82] D. Shreiner, The Khronos OpenGL ARB Working Group y col., *OpenGL programming guide: the official guide to learning OpenGL (Redbook)*. Pearson Education, 2010.

[83] D. Shreiner, *OpenGL(R) Reference Manual: The Official Reference Document to OpenGL (Bluebook)*. Addison Wesley Professional, Boston, 2010.

[84] M. J. Kilgard, *The OpenGL utility toolkit (glut) programming interface (Greenbook)*. Silicon Graphics, 1996.