Memorias del XVI Congreso Latinoamericano de Control Automático, CLCA 2014
Octubre 14-17, 2014. Cancún, Quintana Roo, México

Seguimiento de Trayectorias para un Helicóptero de 4 Rotores AR.Drone 2.0 Utilizando ROS ⋆

J. Santiaguillo-Salinas, E. Aranda-Bricaire

Departamento de Ingeniería Eléctrica, Sección Mecatrónica
CINVESTAV, A.P. 14-740, 07000 México D.F.
(e-mail: jsantiaguillo@cinvestav.mx, earanda@cinvestav.mx).

Resumen: El presente artículo está enfocado al control de seguimiento de trayectorias para helicópteros de 4 rotores; en particular, para el prototipo comercial llamado AR.Drone 2.0 de la empresa francesa Parrot. Con la estrategia de control diseñada se logra que un AR.Drone converja asintóticamente a una trayectoria de vuelo preestablecida. La aplicación de software para el control del AR.Drone se realiza a través de un conjunto de herramientas y librerías proporcionadas por el Sistema Operativo para Robots, ROS por sus siglas en inglés. Se presentan los resultados obtenidos experimentalmente.

Keywords: AR.Drone, Seguimiento de Trayectorias, Helicópteros de 4 Rotores, ROS.

1. INTRODUCCIÓN

Los vehículos aéreos no tripulados (UAV por sus siglas en inglés), son aeronaves que no tienen piloto a bordo. Los UAV pueden ser controlados remotamente por un piloto en tierra o pueden realizar vuelo autónomo a partir de trayectorias pre-programadas. El helicóptero de 4 rotores o quadri-rotor es un tipo de UAV, el cual tiene una configuración en particular, que consta de 4 rotores colocados en las puntas de una estructura en forma de cruz.

El quadri-rotor ha recibido especial atención por parte de los investigadores debido a su facilidad de control comparado con otros vehículos aéreos y por sus capacidades para realizar maniobras agresivas, i.e. giro completo (flip, en inglés). Las principales ventajas de los quadri-rotores son que pueden realizar despegue y aterrizaje vertical y vuelo estacionario (hover), por lo que se pueden utilizar en áreas reducidas; una gran desventaja del quadri-rotor es su tiempo de vuelo, ya que va de 12 a 18 min., por lo que no se pueden realizar vuelos de larga duración.

Durante los últimos años se han presentado algunos trabajos que abordan el diseño y validación experimental de estrategias de control para helicópteros de 4 rotores. González et al. (2013) muestra un control de estabilización de orientación para lograr un vuelo estacionario, Castillo et al. (2004) propone un control basado en el análisis de Lyapunov para realizar despegue, vuelo estacionario y aterrizaje de manera autónoma. Mellinger and Kumar (2011) presenta un control no lineal que asegura el seguimiento de trayectorias y Dryanovski et al. (2013) un sistema de navegación implementado en ROS.

En la literatura se pueden encontrar algunos trabajos referentes al control del AR.Drone en sus versiones 1.0 y 2.0. Wenzel et al. (2012) presenta un esquema líder-seguidor, donde el líder es un AR.Drone controlado desde una aplicación para iPad realizando vuelo de forma autónoma utilizando la odometría visual con la que cuenta. En Jayatilleke and Zhang (2013) se calcula la posición del AR.Drone con respecto a un punto utilizando marcadores para la localización empleando ROS. Jiménez-Lugo and Zell (2013) y Jiménez-Lugo et al. (2013) presentan el control de un AR.Drone modificado, al cual se le agregó un microcontrolador con el que se realiza el sensado y el cálculo del control desde la aeronave, realizando vuelo autónomo sin necesidad de controlarlo remotamente.

El objetivo de este artículo es diseñar y validar experimentalmente una estrategia de control con la finalidad de que un quadri-rotor de la marca Parrot modelo AR.Drone 2.0 logre el seguimiento de trayectorias en el espacio y alcance un ángulo de guiñada deseado (yaw, en inglés).

Este artículo está organizado de la siguiente manera. La Sección 2 presenta los modelos dinámico y cinemático de un helicóptero de 4 rotores, las especificaciones técnicas del AR.Drone 2.0 y una breve explicación de lo que es ROS. El planteamiento del problema se describe en la Sección 3. En la Sección 4 se presenta la estrategia de control para lograr el seguimiento de trayectoria y el análisis de convergencia. Los resultados experimentales obtenidos con la estrategia de control diseñada se muestran en la Sección 5. Finalmente, en la Sección 6 se presentan algunas conclusiones y el trabajo futuro a realizar.

2. PRELIMINARES

2.1 Modelo Dinámico y Cinemático de un Quadri-rotor

La cinemática de un cuerpo rígido que se mueve en el espacio muestra la relación entre las velocidades de traslación y rotación entre el marco del cuerpo y el marco inercial.

⋆ Este trabajo está financiado parcialmente por el CONACyT, México, a través del No. de Becario 243226.

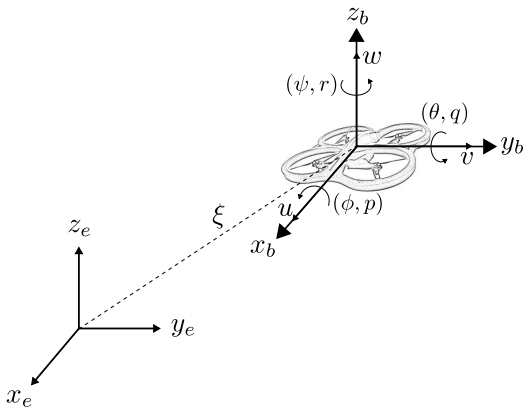

Figura 1. Modelo cinemático del helicóptero de 4 rotores.

De la Figura 1, la posición y rotación del helicóptero de 4 rotores con respecto al marco inercial están dados por $\xi = [x, y, z]^T$ y $\eta = [\phi, \theta, \psi]^T$, respectivamente. Las velocidades lineales del helicóptero dadas en el marco del cuerpo están dadas por $V = [u, v, w]^T$ y $\Omega = [p, q, r]^T$ representan las velocidades angulares. La relación entre las velocidades en el marco inercial y las velocidades lineales en el marco del cuerpo está dada por

$$\dot{\xi} = R_b^e V \tag{1}$$

donde

$$R_b^e = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi c\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix} \tag{2}$$

con $c\phi = \cos\phi$, $s\phi = \sin\phi$, $c\theta = \cos\theta$, $s\theta = \sin\theta$, $c\psi = \cos\psi$ y $s\psi = \sin\psi$. Las velocidades de los ángulos de orientación en el marco inercial y las velocidades angulares en el marco del cuerpo están relacionadas por

$$\Omega = W_\eta \dot{\eta} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \dot{\eta} \tag{3}$$

Invirtiendo (3) obtenemos

$$\dot{\eta} = W_\eta^{-1}\Omega = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & -\sin\phi\sec\theta & \cos\phi\sec\theta \end{bmatrix} \Omega \tag{4}$$

La dinámica representa la evolución de un sistema sujeto a fuerzas y pares externos. Existen varios enfoques para obtener el modelo dinámico de un helicóptero de 4 rotores entre los que se encuentran el enfoque de Newton-Euler, Euler-Lagrange y el de los Cuaterniones. Utilizando el enfoque de Euler-Lagrange y basados en García et at. (2013), el modelo dinámico de un helicóptero de 4 rotores está dado por

$$m\ddot{\xi} = R_b^e \begin{bmatrix} 0 \\ 0 \\ F \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} \tag{5}$$

$$\ddot{\eta} = \tilde{\tau} \tag{6}$$

con m la masa del helicóptero, g la constante de gravedad, F el empuje principal y $\tilde{\tau} = [\tilde{\tau}_\phi, \tilde{\tau}_\theta, \tilde{\tau}_\psi]^T$ torques auxiliares definidos por

Figura 2. Parrot AR.Drone 2.0.

$$\tilde{\tau} = \mathbb{J}^{-1}\left(\tau - C(\eta, \dot{\eta})\dot{\eta}\right) \tag{7}$$

donde $\mathbb{J}$ es la matriz de inercia, $C(\eta_i, \dot{\eta}_i)$ es el termino de Coriolis y τ son los pares generados por los rotores del helicóptero.

2.2 AR.Drone 2.0

El AR.Drone 2.0 (Figura 2) es un helicóptero de 4 rotores comercial, fabricado por Parrot (2013). El AR.Drone 2.0 se puede controlar a través de un dispositivo móvil o smartphone con sistema operativo iOS, Android o Linux, utilizando una conexión Wi-Fi y una aplicación proporcionada por Parrot. Debido a sus especificaciones, el AR.Drone comenzó a tener gran demanda y se comenzaron a desarrollar aplicaciones de control para equipos con arquitectura de PC con sistemas operativos basados en Linux.

El AR.Drone 2.0 cuenta con una estructura con tubos de fibra de carbono, con piezas de plástico nylon con 30 % de fibra de vidrio, 2 cascos de protección de polipropileno expandido, un casco para exteriores y otro para interiores, nanorevestimiento que repele líquidos, así como espuma para aislar el centro inercial de las vibraciones producidas por los motores. En la parte electrónica, el AR.Drone 2.0 cuenta con un procesador ARM Cortex A8 de 1 GHz de 32 bits con Linux 2.6.32, una memoria RAM DDR2 de 1 GHZ, un puerto USB 2.0 que se puede utilizar para grabar video en una memoria USB o conectar un GPS, 2 cámaras de video, una vertical QVGA de 60 FPS y una horizontal HD 720p de 30 FPS.

Los sensores con los que cuenta el AR.Drone 2.0 son:

- Giroscopio de 3 ejes con precisión de 2000º/seg
- Acelerómetro de 3 ejes con precisión de ± 50 mg
- Magnetómetro de 3 ejes con precisión de 6º
- Sensor de presión con una precisión de ± 10 Pa
- Sensor de ultrasonido para medir la posición vertical

El AR.Drone 2.0 tiene 4 motores inrunner sin escobillas de 28,500 RPM y 14.5 W con controladores con CPU AVR de 8 MIPS.

2.3 ROS

El Sistema Operativo para Robots (ROS) es un framework flexible de código abierto, que permite la escritura de software para el control de robots. Se trata de una colección de herramientas, librerías y convenciones que simplifican la creación de comportamientos complejos y robustos en robots. ROS permite el diseño de manera individual de procesos que se pueden acoplar de manera flexible en tiempo de ejecución. Estos procesos se pueden agrupar en paquetes que pueden ser fácilmente compartidos y distribuidos, con lo que se logra la reutilización de código.

ROS proporciona los servicios que se esperan de un sistema operativo, incluyendo abstracción del hardware, control de dispositivos a bajo nivel, envío de mensajes entre procesos y administración de paquetes. También proporciona herramientas y librerías para la obtención, construcción, escritura y ejecución de código en varios equipos. Actualmente, ROS sólo puede ser ejecutado en plataformas basadas en Linux.

Algunas plataformas robóticas que pueden utilizar ROS son Aldebaran Nao, Lego NTX, iRobot Roomba, Merlin miabotPro, AscTec Pelican y HummingBird Quadrotors, Festo Didactic Robotino, todos los robots de Adept MobileRobots (ActivMedia), Allegro Hand SimLab, Parrot AR.Drone, entre otras. Para mayor información sobre como utilizar ROS y conocer más plataformas robóticas en las que se puede emplear, referirse a ROS (2013).

3. PLANTEAMIENTO DEL PROBLEMA

Sea $m(t) = [m_x(t), m_y(t), m_z(t)]^T$ una trayectoria preestablecida la cual se supone dos veces continuamente diferenciable. Sea ψ_d un ángulo de guiñada deseado (yaw). El objetivo de este trabajo es diseñar una estrategia de control para un AR.Drone 2.0 tal que

- Se logre el seguimiento asintótico de una trayectoria preestablecida, i.e.
$$\lim_{t\to\infty} (\xi(t) - m(t)) = 0.$$
- Se alcance un ángulo de orientación deseado en yaw, i.e.
$$\lim_{t\to\infty} (\psi(t) - \psi_d(t)) = 0.$$

4. ESTRATEGIA DE CONTROL

Debido a la arquitectura del AR.Drone 2.0, sólo se pueden controlar los ángulos de alabeo y cabeceo (roll y pitch), con lo que se logra el movimiento del quadri-rotor en el plano XY; la velocidad w, con lo que se logra el desplazamiento en el eje Z; y la velocidad r del drone, con lo que se logra la rotación del quadri-rotor sobre el eje Z (ángulo de guiñada). Conociendo lo anterior, se propone una estrategia de control lineal a partir de los modelos presentados en la Sección 2.

Linealizando (5) alrededor de $\xi = \xi_d(t)$, $\dot{\xi} = 0$, $\phi = \theta = 0$, $\psi = \psi_d(t)$, $\dot{\phi}, \dot{\theta}, \dot{\psi} = 0$ con entradas nominales dadas por $F = mg$ y $\tilde{\tau} = 0$ como en Mahony et al. (2012) y considerando sólo la dinámica en XY obtenemos

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = gA(\psi_d) \begin{bmatrix} \phi_d \\ \theta_d \end{bmatrix} = g \begin{bmatrix} \sin\psi_d & \cos\psi_d \\ -\cos\psi_d & \sin\psi_d \end{bmatrix} \begin{bmatrix} \phi_d \\ \theta_d \end{bmatrix} \quad (8)$$

Definiendo variables de control auxiliares $r = [r_x, r_y]^T$ es posible establecer una estrategia para controlar la posición en X y Y dada por

$$\begin{bmatrix} \phi_d \\ \theta_d \end{bmatrix} = \frac{1}{g} A^{-1}(\psi_d) \begin{bmatrix} r_x \\ r_y \end{bmatrix} = \frac{1}{g} \begin{bmatrix} \sin\psi_d & -\cos\psi_d \\ \cos\psi_d & \sin\psi_d \end{bmatrix} \begin{bmatrix} r_x \\ r_y \end{bmatrix} \quad (9)$$

donde

$$\begin{bmatrix} r_x \\ r_y \end{bmatrix} = \ddot{m}(t) - \begin{bmatrix} k_{dx}(\dot{x} - \dot{m}_x(t)) + k_{px}(x - m_x(t)) \\ k_{dy}(\dot{y} - \dot{m}_y(t)) + k_{py}(y - m_y(t)) \end{bmatrix} \quad (10)$$

con $k_{dx}, k_{px}, k_{dy}, k_{py}$ ganancias de control. Las señales ϕ_d y θ_d son las que se enviarán al AR.Drone 2.0 para controlar su posición en el plano XY. A partir de la linealización anterior, también obtenemos que

$$\dot{z} = w \quad (11)$$
$$\dot{\psi} = r \quad (12)$$

Proponemos

$$w = -k_{pz}(z - m_z(t)) + \dot{m}_z(t) \quad (13)$$
$$r = -k_{p\psi}(\psi - \psi_d(t)) + \dot{\psi}_d(t) \quad (14)$$

donde $k_{pz}, k_{p\psi}$ son ganancias de control, $\dot{z}$ es la señal para controlar el desplazamiento del AR.Drone 2.0 en el eje Z y $\dot{\psi}$ es la señal para controlar la rotación de guiñada.

Los resultados principales de este trabajo, recordando que son resultados locales ya que se obtienen a partir de una linealización, son los siguientes

Teorema 1. Considere el sistema (8) y la estrategia de control (9). Suponga que k_{dx}, $k_{px} > 0$ y k_{dy}, $k_{py} > 0$. Entonces, en el sistema en lazo cerrado (8)-(9), el AR.Drone 2.0 converge localmente a la trayectoria deseada en el plano XY, i.e. $\lim_{t\to\infty}(x(t) - m_x(t)) = 0$ y $\lim_{t\to\infty}(y(t) - m_y(t)) = 0$.

Demostración. El sistema en lazo cerrado (8)-(9) produce

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = A \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} + B \begin{bmatrix} x \\ y \end{bmatrix} + C(t) \quad (15)$$

donde

$$A = \begin{bmatrix} -k_{dx} & 0 \\ 0 & -k_{dy} \end{bmatrix} \quad , \quad B = \begin{bmatrix} -k_{px} & 0 \\ 0 & -k_{py} \end{bmatrix} \quad (16)$$

$$C(t) = \begin{bmatrix} \ddot{m}_x(t) + k_{dx}\dot{m}_x(t) + k_{px}m_x(t) \\ \ddot{m}_y(t) + k_{dy}\dot{m}_y(t) + k_{py}m_y(t) \end{bmatrix} \quad (17)$$

Definimos los errores del sistema en el plano XY como

$$e_x = x - m_x(t), \quad (18)$$
$$e_y = y - m_y(t). \quad (19)$$

En forma matricial tenemos

$$\begin{bmatrix} e_x \\ e_y \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} - \begin{bmatrix} m_x(t) \\ m_y(t) \end{bmatrix}. \quad (20)$$

La dinámica del error en el plano XY está dada por

$$\begin{bmatrix} \ddot{e}_x \\ \ddot{e}_y \end{bmatrix} = A \begin{bmatrix} \dot{e}_x \\ \dot{e}_y \end{bmatrix} + B \begin{bmatrix} e_x \\ e_y \end{bmatrix} \quad (21)$$

Definiendo

$$\lambda = [e_x \quad \dot{e}_x \quad e_y \quad \dot{e}_y]^T \quad (22)$$

podemos reescribir la dinámica del error como

$$\dot{\lambda} = M\lambda \quad (23)$$

donde

$$M = \begin{bmatrix} N_1 & 0 \\ 0 & N_2 \end{bmatrix} \tag{24}$$

con

$$N_1 = \begin{bmatrix} 0 & 1 \\ -k_{px} & -k_{dx} \end{bmatrix} \quad , \quad N_2 = \begin{bmatrix} 0 & 1 \\ -k_{py} & -k_{dy} \end{bmatrix} \tag{25}$$

La matriz M es diagonal a bloques por lo que el análisis se reduce al estudio de las matrices N_1 y N_2 que son matrices de 2 dimensiones. De esta forma, para cada ley de control con $k_{px}, k_{dx} > 0$ y $k_{py}, k_{dy} > 0$ la dinámica en el plano XY es localmente asintóticamente estable. ■

Teorema 2. Considere el sistema (11)-(12) y la estrategia de control (13)-(14). Suponga que k_{pz}, $k_{p\psi} > 0$. Entonces, en el sistema en lazo cerrado (11)-(12)-(13)-(14), el AR.Drone 2.0 converge localmente a la trayectoria deseada en el eje Z, i.e. $\lim_{t\to\infty}(z(t) - m_z(t)) = 0$ y a la orientación deseada de guiñada, i.e. $\lim_{t\to\infty}(\psi(t) - \psi_d(t)) = 0$.

Demostración. El sistema en lazo cerrado (11)-(12)-(13)-(14) produce

$$\begin{bmatrix} \dot{z} \\ \dot{\psi} \end{bmatrix} = D \begin{bmatrix} z \\ \psi \end{bmatrix} + E(t) \tag{26}$$

donde

$$D = \begin{bmatrix} -k_{pz} & 0 \\ 0 & -k_{p\psi} \end{bmatrix} \quad , \quad E = \begin{bmatrix} \dot{m}_z(t) + k_{pz} m_z(t) \\ \dot{\psi}_d(t) + k_{p\psi}\psi_d(t) \end{bmatrix} \tag{27}$$

Definimos los errores para Z y el ángulo de guiñada como

$$e_z = z - m_z(t) \tag{28}$$

$$e_\psi = \psi - \psi_d(t) \tag{29}$$

en forma matricial tenemos

$$\begin{bmatrix} e_z \\ e_\psi \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} z \\ \psi \end{bmatrix} - \begin{bmatrix} m_z(t) \\ \psi_d(t) \end{bmatrix} \tag{30}$$

La dinámica del error en Z y en el ángulo de guiñada resulta

$$\begin{bmatrix} \dot{e}_z \\ \dot{e}_\psi \end{bmatrix} = D \begin{bmatrix} e_z \\ e_\psi \end{bmatrix} \tag{31}$$

Claramente, la matriz D es Hurwitz y los errores convergen localmente asintóticamente a cero. ■

5. RESULTADOS EXPERIMENTALES

Para validar los resultados teóricos obtenidos, la estrategia de control diseñada se implementa en una plataforma experimental. La plataforma se muestra en la Figura 3 y está formada por un quadri-rotor Parrot AR.Drone 2.0, 12 cámaras y 2 computadoras tipo PC.

La localización del AR.Drone 2.0 (posición y orientación) se realiza a través de un sistema de visión que consta de 12 cámaras marca OptiTrack modelo Flex 13 y una PC Core i3 con Windows 7 que tiene el software Motive:Tracker, dicho software está diseñado para el rastreo de objetos en 6 DOF con alta precisión y con soporte para el procesamiento en tiempo real. Las velocidades del AR.Drone 2.0 requeridas en la ley de control para el desplazamiento en el plano XY se obtienen a partir de las mediciones de los sensores con los que cuenta el helicóptero; sino se contara

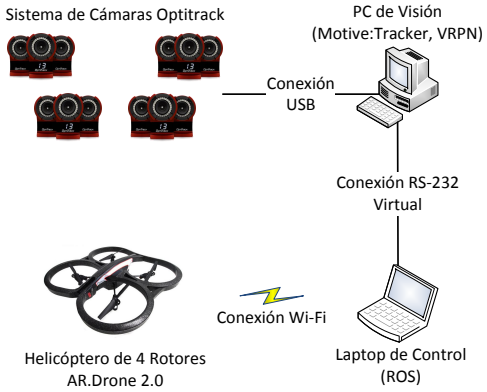

Figura 3. Plataforma experimental.

con tales sensores, se optaría por estimar la velocidad utilizando observadores.

Se cuenta con una Laptop Core i5 con Ubuntu 12.04 con ROS, que realiza los cálculos del control y los envía al AR.Drone 2.0 vía Wi-Fi. La comunicación entre la PC de visión y la Laptop de control se hace a través de una transmisión VRPN mediante una conexión serial virtual por un puerto USB.

Ya que los experimentos se realizan en un espacio cerrado, se utiliza el casco de protección para interiores del AR.Drone 2.0, con el cual el helicóptero tiene un peso de 0.42 kg. Se realizan 3 experimentos donde un AR.Drone 2.0 debe seguir una trayectoria preestablecida manteniendo un cierto ángulo de guiñada constante. Los parámetros usados en los experimentos son $k_{dx} = k_{px} = k_{dy} = k_{py} = k_{pz} = 2$ y $k_{p\psi} = 6$. La trayectoria deseada en el plano XY es una Lemniscata de Gerono dada por

$$m_x(t) = 1.4\cos\left(\frac{2\pi t}{T}\right) \quad , \quad m_y(t) = 0.7\sin\left(\frac{4\pi t}{T}\right) \tag{32}$$

y la trayectoria deseada en Z por

$$m_z(t) = 1 + 0.25\cos\left(\frac{2\pi t}{T} + \pi\right) \tag{33}$$

con un periodo de $T = 30$ s.

Experimento 1. El AR.Drone 2.0 mantiene un ángulo de yaw deseado $\psi_d = 0^o$. La Figura 4 muestra el movimiento del AR.Drone en el espacio. La Figura 5 y la Figura 6 muestran los errores de posición y orientación, respectivamente.

Experimento 2. El AR.Drone 2.0 mantiene un ángulo de yaw deseado $\psi_d = 45^o$. La Figura 7 muestra el movimiento del AR.Drone en el espacio. La Figura 8 y la Figura 9 muestran los errores de posición y orientación, respectivamente.

Experimento 3. El AR.Drone 2.0 mantiene un ángulo de yaw deseado $\psi_d = 90^o$. La Figura 10 muestra el movimiento del AR.Drone en el espacio. La Figura 11 y la Figura 12 muestran los errores de posición y orientación, respectivamente.

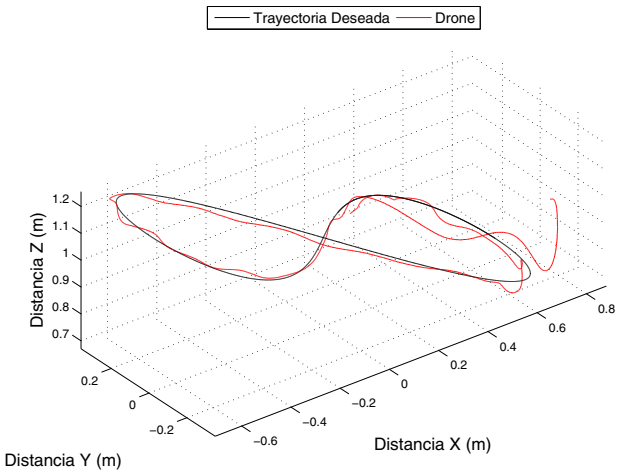

Figura 4. Movimientos del drone en el espacio, $\psi_d = 0^o$.

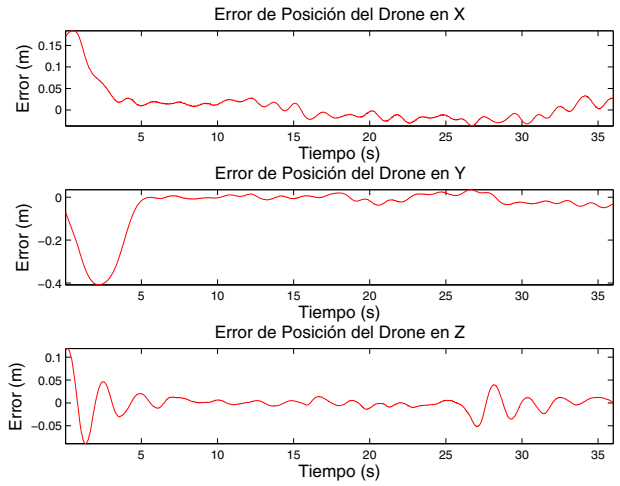

Figura 5. Errores de posición del drone para $\psi_d = 0^o$.

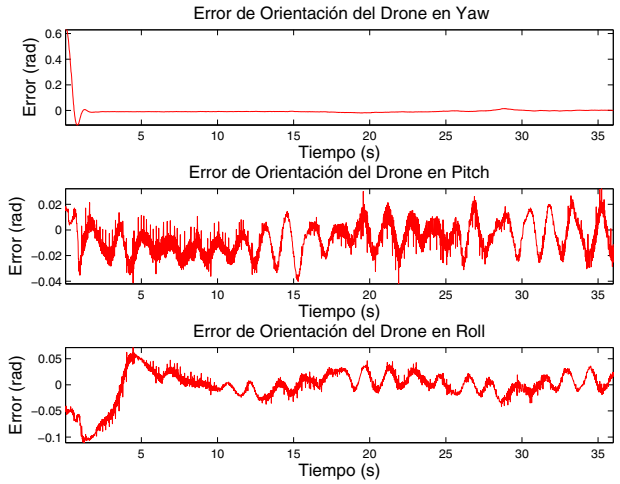

Figura 6. Errores de orientación del drone para $\psi_d = 0^o$.

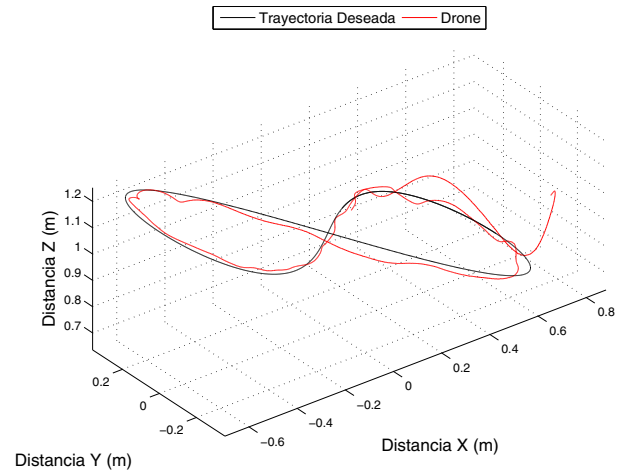

Figura 7. Movimientos del drone en el espacio, $\psi_d = 45^o$.

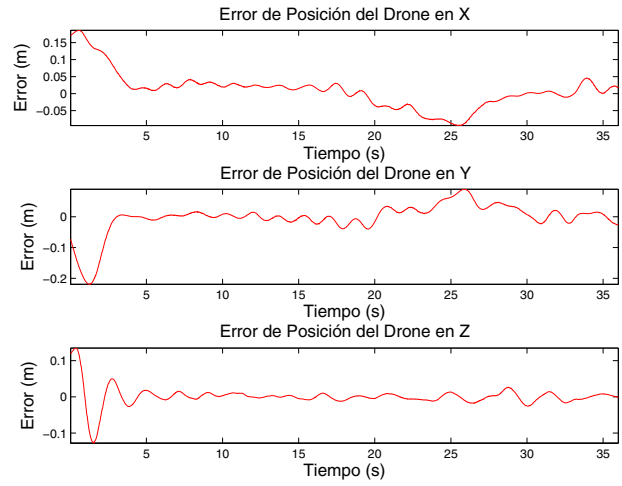

Figura 8. Errores de posición del drone para $\psi_d = 45^o$.

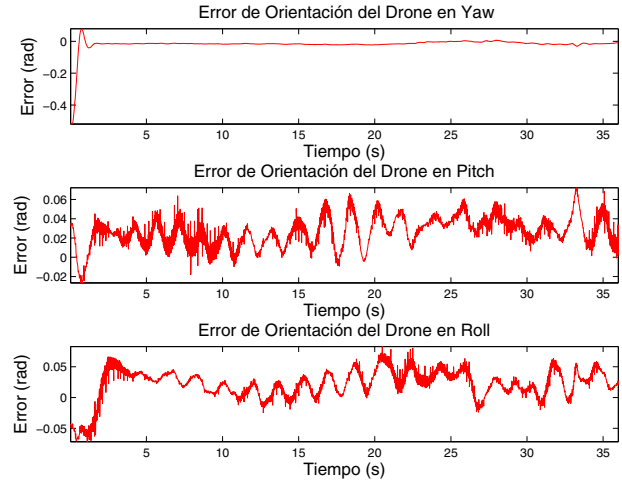

Figura 9. Errores de orientación del drone para $\psi_d = 45^o$.

Cabe señalar que, para el despegue y aterrizaje del AR.Drone, se utilizaron las rutinas implementadas en la aeronave. También hay que mencionar que las señales de control fueron escalas a un rango de [-1, 1], ya que estos son los valores de entrada que recibe el AR.Drone, donde ± 1 son los valores máximos y mínimos tanto para los ángulos de alabeo y cabeceo (en radianes), como para las velocidades lineal (en mm/s) y rotacional (en rad/s) en Z. Estos valores se especifican en un archivo donde se configuran los parámetros del AR.Drone. Para verificar las diferencias que pueden existir al utilizar ROS y al no utilizarlo para el control del AR.Drone 2.0, se puede hacer

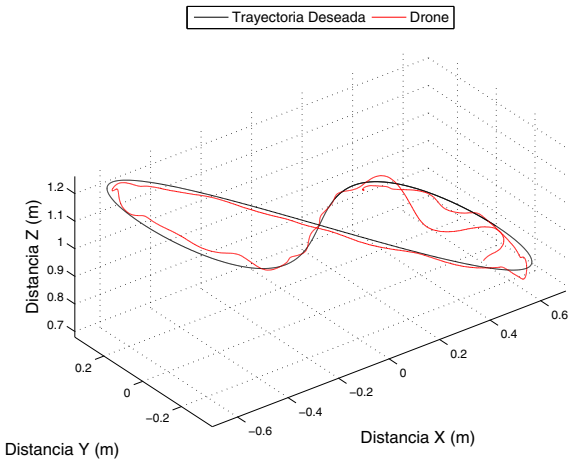

Figura 10. Movimientos del drone en el espacio, $\psi_d = 90^o$.

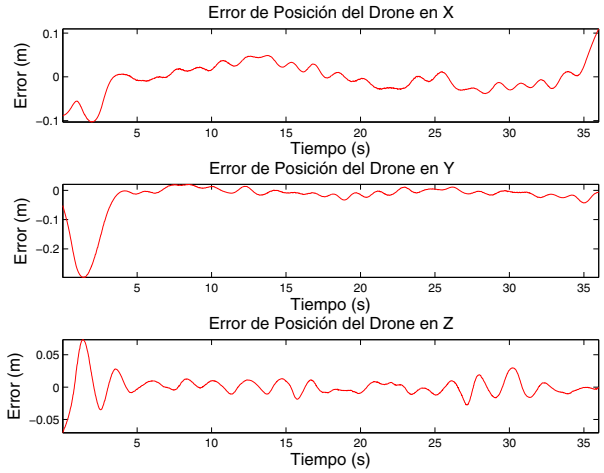

Figura 11. Errores de posición del drone para $\psi_d = 90^o$.

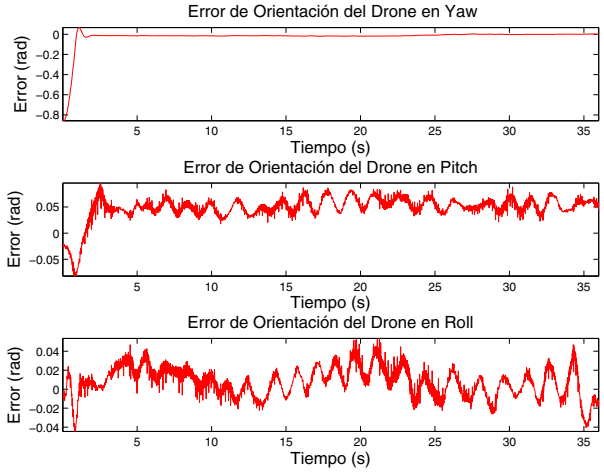

Figura 12. Errores de orientación del drone para $\psi_d = 90^o$.

una comparación entre los resultados presentados en este trabajo y los resultados presentados en Rosaldo-Serrano and Aranda-Bricaire (2014).

6. CONCLUSIONES Y TRABAJO FUTURO

Este artículo presenta una estrategia de control para lograr el seguimiento de trayectorias en el espacio por parte de un helicóptero de 4 rotores AR.Drone 2.0. Como se puede observar en los resultados experimentales obtenidos, se logra el seguimiento de trayectorias con un ángulo de orientación deseado de guiñada. Para este tipo de vehículo, el error observado puede considerarse aceptable y se debe a efectos no lineales que se presentan debido a desbalanceo y deformaciones en las hélices, ejes y engranes de la aeronave. Además, hay que recordar que los resultados son obtenidos a partir de una linealización local.

Como trabajo futuro, se plantea el uso de varios helicópteros de 4 rotores para formar un sistema multi-agente y lograr vuelo en formación. Asimismo, implementar una estrategia de control no lineal para mejorar el desempeño de la aeronave y obtener resultados mas generales.

REFERENCIAS

Castillo, P., Lozano, R. and Dzul, A. (2004). Stabilization of a mini-rotorcraft having four rotors. *In proc. of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3, 2693–2698.

Dryanovski, I., Valenti, R.G. and Xiao, J. (2013). An open-source navigation system for micro aerial vehicles. *Auton. Robot.*, 34(3), 177–188. Springer, US.

García, L.R., Dzul, A.E., Lozano, R. and Pégard, C. (2013). Quad rotorcraft control. *Advances in Industrial Control*. Springer, London.

González, I., Salazar, S., Torres, J., Lozano, R. and Romero, H. (2013). Real-Time attitude stabilization of a mini-UAV quad-rotor using motor speed feedback. *J. Intell. Robot. Syst.*, 70(1-4), 93–106. Springer, Netherlands.

Jayatilleke, L. and Zhang, N. (2013). Landmark-based localization for unmanned aerial vehicles. *2013 IEEE International Systems Conference (SysCon)*, 448–451.

Jiménez-Lugo, J., Masselli, A., and Zell, A. (2013). Following a quadrotor with another quadrotor using onboard vision. *2013 European Conference on Mobile Robots (ECMR)*, 26–31.

Jiménez-Lugo, J. and Zell, A. (2013). Framework for autonomous onboard navigation with the AR.Drone. *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, 575–583.

Mahony, R., Kumar, V. and Corke, P. (2012). Multirotor aerial vehicles: modeling, estimation, and control of quadrotor. *IEEE Robot. Autom. Mag.*, 19(3), 20–32.

Mellinger, D. and Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. *2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2520–2525.

Parrot AR.Drone 2.0 (2013). *http://ardrone2.parrot.com/*.

ROS (2013). *http://www.ros.org/*.

Rosaldo-Serrano, M.A. and Aranda-Bricaire, E. (2014). Modelado Y Control De Un Prototipo Comercial De Aeronave Tipo Quadrirotor. *XVI Congreso Latinoamericano de Control Automático CLCA 2014*. Cancún, Quintana Roo, México.

Wenzel, K.E., Masselli, A. and Zell, A. (2012). Visual tracking and following of a quadrocopter by another quadrocopter. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 4993–4998.