



Integración de tecnologías en el control de navegación para el Cuadricóptero Parrot AR Drone 2.0

Méndez Navarro Alejandro✉, Rodríguez Hernández José Jorge, Villalobos Guerra Arnoldo Ulises y Jorge Hugo Ávalos García

Instituto Tecnológico de Estudios Superiores de Zamora

✉amendez@teczamora.mx

Resumen

El AR Drone 2.0 es un vehículo aéreo no tripulado que permite al usuario conocer su sistema tanto de software como el hardware, conociendo la parte del software podemos agregar o quitar algunas de las características propias del cuadricóptero, así como crear alguna aplicación con interfaz gráfica en puede ser manipulado por el usuario. El cuadricóptero AR Drone 2.0 fue diseñado para trabajar principalmente con el sistema operativo MAC, pero en la actualidad ya existen programas y compiladores para trabajar con Linux o con Microsoft aunque resulta complicado la comunicación entre la PC y el Drone además de conocer la función de los comandos AT con los cuales el Drone interpreta las órdenes del usuario. En este artículo se desarrolla un control de navegación para el Cuadricoptero Parrot AR Drone 2.0 el cual pueda ser utilizado para la integración de tecnologías, tales como mediciones de variables físicas.

Palabras clave: Cuadricóptero, integración de tecnologías, control de navegación, control automático, variables de estado.

Abstract

The AR Drone 2.0 is an unmanned aerial vehicle that allows the user to know their system both software and hardware, knowing the software part we can add or remove some of the characteristics of the quadcopters, as well as create some application with graphical interface in It can be manipulated by the user. The AR Drone 2.0 Quadcopter was designed to work mainly with the MAC operating system, but at present there are already programs and compilers to work with Linux or Microsoft, although the communication between the PC and the Drone is complicated, as well as knowing the function of the AT commands with which the Drone interprets the user's commands. In this article a navigation control is developed for the Parrot AR Drone 2.0 Quadriopter which can be used for the integration of technologies, such as measurements of physical variables.

Keywords: Quadcopters, technology integration, navigation control, automatic control, state variables.

1. Introducción

Hoy en día los robots se encuentran en distintas áreas, desde la producción masiva de cualquier producto industrial, en exploración espacial y marítima, industria bélica y hasta en el deseo de sintetizar algunos aspectos de la función humana mediante el uso de mecanismos, sensores, actuadores y computadoras; tienen la capacidad de desarrollar múltiples tareas de manera flexible según su programación. El Parrot AR Drone 2.0, ver figura 1, es un cuadricóptero de control remoto de la empresa francesa Parrot SA, es controlado por el usuario a través de una aplicación que está actualmente disponible para iOS y Android, existe un software no oficial disponible para Windows

Phone [1]. La segunda versión de la AR Drone se dio a conocer en el *Consumer Electronics Show* (CES, por sus siglas en inglés) en la ciudad Las Vegas en 2012 [1-2].



Figura 1. Cuadricóptero Parrot AR Drone 2.0.

La llegada de los vehículos aéreos no tripulados comerciales dio pauta a que muchas personas se interesaran por el desarrollo de aplicaciones, mejoramientos y una gran cantidad de proyectos relacionados, además que varias compañías desarrollaran una gran diversidad de modelos para que los aficionados y expertos pudieran jugar con ellos o tener un control total. En la Tabla 1 se mencionan algunos de los tipos de vehículos aéreos no tripulados existentes en la actualidad.

Tabla 1. Tipos de Vehículos no tripulados.

| | |
|--------|----------------------------|
| Drones | Reconocimiento |
| | Combate |
| | Investigación y desarrollo |

Dado a la implementación constante de nuevas tecnologías como método de enseñanza e investigación tanto por parte de una comunidad estudiantil y personas con ansias de aprendizaje, el modelo de cuadricóptero Parrot AR Drone 2.0 permite un fácil control del mismo por medio de una interfaz móvil, además de contar con un *Software Development Kit* (SDK, por sus siglas en inglés) orientada al desarrollo de aplicaciones. El SDK es el punto fuerte en el cual se basa este modelo, no existen demasiados cuadricópteros comerciales que dispongan de un método de desarrollo de aplicaciones, ya que para algo más sofisticado se tiene que migrar a plataformas de microcontroladores [1-8].

El uso de un SDK para realización de proyectos para un cuadricóptero puede resultar un tanto engorroso para algunos usuarios que tengan escaso o nulo conocimientos de programación o sistemas operativos, ya que este modelo fue especialmente diseñado para una manipulación en el sistema operativo Linux, en este sistema operativo se encuentran con un sin número de proyectos dado que el cuadricóptero AR Drone 2.0 cuenta con plataformas de códigos libres y es de fácil acceso [1-4]. Sin embargo, los problemas mencionados anteriormente en el presente párrafo se repiten, por lo que en base a esta problemática se realizó una investigación de una manera de programación o manipulación del cuadricóptero en un sistema operativo más sencillo, como el caso de Windows [9].

El software Node.JS es un entorno en tiempo de ejecución que permite por medio de comandos a llamar archivos java script el cual es una forma de código amistoso y sencillo, además de poder lograr una comunicación con el Drone permitiéndonos manipular el código por medio de comandos AT propios del Drone, desde el control del despegue, aterrizaje, movimientos, control de luces, diversas maniobras, visualización de la cámara y despliegue de un stream en tiempo real en el monitor de la PC hasta el punto de llegar a programar distintas rutinas y aplicaciones como el seguimiento facial o de objetos o trabajar en la navegación del cuadricóptero [5-9].



La navegación se puede definir como el proceso de dirigir los movimientos de un vehículo de un punto a otro: siguiendo una línea negra en el suelo, moviéndose a través de un laberinto, siguiendo una pared o siguiendo un camino aleatorio [1]. El robot móvil reacciona a las variables y parámetros que se encuentran directamente a su entorno tales como: intensidad de luz, posición relativa respecto a una línea blanca o negra, debido al contacto con una pared y demás. El robot móvil realiza movimientos aleatorios de sensado para identificar a los obstáculos; la mejor manera de identificar los obstáculos es representando la toma de decisiones teniendo en cuenta el conocimiento total o parcial del entorno [1-9].

2. Plataforma del cuadricóptero Parrot AR Drone 2.0

2.1 Componentes mecánicos

El cuadricóptero Parrot AR Drone 2.0 consta de material plástico y espuma y mide unos 30 cm. La conexión se realiza usando Wi-Fi y es controlable por smartphones y tabletas usando iOS O los sistemas operativos de Android, que permite la transmisión del flujo de vídeo de ambas cámaras. Las características incluyen el seguimiento de objetos y la compatibilidad para las aplicaciones AR-Juego [2]. Se puede decir que tiene una estructura ligera, ya que su peso máximo es de 420 gramos con la carcasa incluida. Eso y el alcance de la conexión Wi-Fi le permiten llegar a altitudes de hasta 50 - 120 metros (dependiendo de las condiciones climáticas) y alcanzar velocidades de hasta 18 km/h. [3]

2.2 Componentes eléctricos

Su estructura central está formada por el cuerpo que contiene los elementos más significativos del dispositivo. Tiene una batería de litio recargable que permite una autonomía de vuelo de aproximadamente 15 minutos (según condiciones).

Este tipo de cuadricópteros, normalmente manejados a control remoto, pese a que puedan parecer simples juguetes, poseen complejos sistemas de control que les permiten mantener un vuelo estable y bien definido, además de incluir diversos sensores en base a la tarea para la que sean diseñados.

Con respecto a los sensores, el AR Drone dispone de:

- Sensor de ultrasonidos para medir los cambios de altitud.
- Sensor de altitud para correcciones.
- Acelerómetro digital de tres ejes para monitorizar los movimientos de posición.
- Giróscopo de dos ejes y giróscopo preciso piezoeléctrico para medir el giro y en cabeceo.
- Magnetómetro de tres ejes para la orientación.
- Todos estos sensores están impresos en un circuito montado sobre un esqueleto de fibra de carbono. En su totalidad resultan cruciales para determinar la posición y orientación del Drone en el vuelo. [3]

El sistema informático integrado consiste en un microprocesador ARM9 RISC de 32 bits @ 468 MHz, una memoria DDR SDRAM de 128 MB, un sistema operativo con núcleo Linux, un modem Wi-Fi b/g y además dispone de un conector USB de alta velocidad para poder almacenar los vídeos y las fotos que se realicen [1-9].

2.3 Comunicación

Con respecto a la comunicación del AR Drone con los diferentes dispositivos, la transmisión se realiza cuando éstos se conectan a una red inalámbrica en el cuadricóptero. Para controlar y configurar el AR Drone se utilizan los AT Commands, que se transfieren por el puerto UDP 5556 con una frecuencia de 30 Hz [3].

3. Metodología

El cuadricóptero puede volar a cualquier punto dado en el espacio ya sea por una secuencia de movimientos diferentes o por la combinación de varios movimientos juntos. El movimiento se logra dando valores de referencia como entrada a los controladores internos, como se indica en la figura 2 [10].



Figura 2. Bloque de control para AR Drone 2.0.

Los movimientos aéreos en el cuadricóptero son similares a los de un helicóptero convencional, la diferencia es que el movimiento se varía cada una de las cuatro velocidades del motor para obtener un efecto deseado que causa movimiento. Los movimientos del cuadricóptero se explican en la Tabla 2.

Tabla 2. Movimientos del cuadricóptero Parrot AR Drone 2.0

| | |
|------------|---|
| Pitch | Movimiento de rotación a lo largo del eje transversal y generando un movimiento de traslación en el eje x. |
| Roll | Movimiento de rotación a lo largo del eje longitudinal x generando un movimiento de traslación en el eje y. |
| Yaw | Movimiento de rotación a lo largo del eje z. El cuadricóptero gira alrededor de su eje vertical. |
| Acelerador | Movimiento traslaciones sobre el eje z. El cuadricóptero asciende o desciende. |

Un modelo matemático de un cuadricóptero se derivó de los primeros principios e implementado en Simulink [10] bajo las siguientes características:

1) El cuadricóptero es un sistema bajo-accionado, tiene seis grados de libertad (asumiendo la dinámica rígida del cuerpo [10]) pero solamente cuatro entradas. La aceleración lateral es un resultado indirecto de una actitud distinta de cero y no puede ser controlada directamente.

2) Las cuatro entradas se pueden definir en términos de sumas y diferencias de velocidades del rotor. Esto permite que las ecuaciones de movimiento se escriban en cuatro ecuaciones independientes que gobiernan la velocidad de balanceo, la velocidad de oscilación, la velocidad de guiñada y la velocidad vertical.

3) El cuadricóptero es un sistema inherentemente inestable y requiere un control adecuado para lograr un movimiento útil. Esta inestabilidad inherente, sin embargo, también conduce a una mayor maniobrabilidad.



Un cuadricóptero es un dispositivo que tiene cuatro rotores dispuestos, ver Figura 2. Las fuerzas y momentos debidos a la rotación de los rotores controlan directamente la aceleración angular y la aceleración vertical relativa del cuerpo del cuadricóptero y por lo tanto indirectamente controlan la aceleración lateral. Tanto el paso como el eje de rotación de los rotores se fijan típicamente.

Se derivó un modelo no lineal, linealizado alrededor de la condición de hover y luego expresado en forma de espacio de estado. El modelo fue linealizado para permitir el análisis y control del sistema LTI (Linear Time-Invariant). El modelo lineal es lo suficientemente preciso ya que el cuadricóptero no alcanzará una actitud mayor que 30° , por lo tanto, no presentará un desvío significativo del estado de hover.

Las ecuaciones derivadas a continuación son similares a las presentadas por el GRASP Lab [10] y la China Jiliang University [11]. El formulismo de Newton-Euler fue utilizado, a diferencia del Lagrangian, debido a las ecuaciones intuitivas y reconocibles ($F = ma$ y $\tau = I\alpha$). El modelo no considera efectos de orden superior tales como resistencia aerodinámica, aleteo de la cuchilla o par giroscópico tal como en el documento de Huang et. Alabama. [12] ya que no se requiere un modelo de alta fidelidad.

Se definen dos marcos de referencia: un marco inercial en el que se escriben las ecuaciones de Newton y un marco no inercial en el que escribir las ecuaciones de Euler. Esto fue por conveniencia ya que las ecuaciones de Euler no necesitan ser escritas en un marco inercial y serían incómodas e intuitivas si lo fueran.

El marco inercial fue dado el subíndice W , para el mundo, o referido por $\{W\}$. Del mismo modo, el subíndice B , y $\{B\}$, se dieron al cuerpo fijo no inercial cuyo origen está en el CoM del cuadricóptero y cuyos ejes giran con el cuadricóptero [15-22].

El origen de $\{B\}$ a $\{W\}$ relativo a se describió usando el vector de posición \mathbf{r} . Todos estos términos se pueden ver en la Figura 3. Debido a que las coordenadas del marco inercial se usan a lo largo de las siguientes ecuaciones matemáticas, para mayor claridad no se les da el subíndice W . Donde las coordenadas no están en $\{W\}$, esto queda claro.

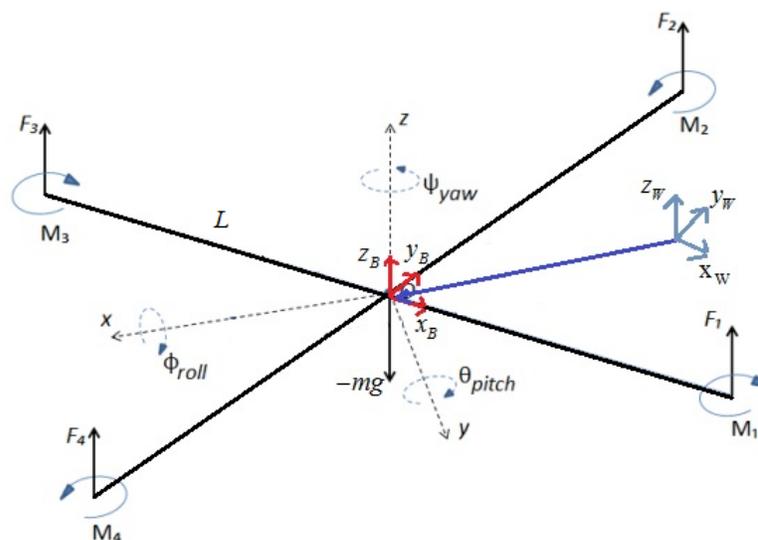


Figura 3. Marcos de referencia en el cuadricóptero.



La matriz rotacional R se definió para transformar coordenadas $\{B\}$ en $\{W\}$ coordenadas usando los ángulos de Euler ϕ, θ y ψ , y que se refieren al balanceo, paso y guiñada, respectivamente. Los ángulos se definen de la siguiente manera: para obtener de $\{W\}$ a $\{B\}$ uno rota alrededor de $z_B (= z_W)$ por ψ , después $x_B (\neq x_W)$ por ϕ , y finalmente y_B alrededor de θ [13]. Formalmente esto se define como los ángulos de Euler de Z-X-Y y la matriz de rotación es (1) [15-22].

$$R = \begin{bmatrix} c\psi c\theta - n\phi s\psi s\theta & -c\phi s\psi & c\psi s\theta + c\theta s\phi s\psi \\ c\theta s\psi + c\psi s\phi s\theta & c\phi c\psi & s\psi s\theta - c\psi c\theta s\phi \\ -c\phi s\theta & s\phi & c\phi c\theta \end{bmatrix} \quad (1)$$

La ecuación de Newton [17-20], que sólo se mantiene en el marco inercial, está por debajo (2). La gravedad actúa en la dirección de $-\hat{z}_W$ y las fuerzas de los rotores actúan en la dirección \hat{z}_B que se transforma en la matriz $\{W\}$ rotacional.

$$m\ddot{\mathbf{r}} = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + R \begin{bmatrix} 0 \\ 0 \\ \sum F_i \end{bmatrix} = \begin{bmatrix} \sum F_i (c\psi s\theta + c\theta s\phi s\psi) \\ \sum F_i (s\psi s\theta - c\psi c\theta s\phi) \\ \sum F_i (c\phi c\theta) - mg \end{bmatrix} \quad (2)$$

Las ecuaciones de movimiento de rotación se formulan en $\{B\}$ más convenientemente ya que el tensor de inercia es invariante en el tiempo y la simetría del cuerpo puede usarse para simplificar ecuaciones [18-22]. La ecuación de Euler, $\mathbf{I}\dot{\boldsymbol{\omega}} = \mathbf{M} - \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}$ se aplicó al cuadricóptero y se derivaron las dinámicas de rotación. Las ecuaciones se dan en la ecuación (3). Dado que el origen de $\{B\}$ se define como el CoM del cuadricóptero, la ley de inercia de Sylvester establece que los productos de inercia (aquellos términos que no están en la diagonal) son cero. Y como el cuadricóptero es aproximadamente simétrico alrededor de x y y , $I_{xx} \approx I_{yy}$. El tensor de inercia es entonces una matriz diagonal y las ecuaciones pueden ser simplificadas [15-22].

Dinámica de rotación

$$\mathbf{I} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} L(F_2 - F_4) \\ L(F_3 - F_1) \\ M_1 - M_2 + M_3 - M_4 \end{bmatrix} - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \mathbf{I} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad (3)$$

Dinámica de rotación simplificada

$$\begin{aligned} I_{xx}\dot{p} &= Lk_F(\omega_2^2 - \omega_4^2) - qr(I_{zz} - I_{yy}) \\ I_{yy}\dot{q} &= Lk_F(\omega_3^2 - \omega_1^2) - pr(I_{xx} - I_{zz}) \\ I_{zz}\dot{r} &= k_M(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{aligned} \quad (4)$$



3.1 Modelo linealizado

Muchas instituciones, incluyendo el GRASP Lab [9-14], utilizan un modelo de motor simplificado en el que la fuerza ascendente y el par reactivo son proporcionales al cuadrado de la velocidad del rotor y la respuesta se rige por una ecuación diferencial de primer orden (5). El subíndice diferencia entre cada motor / rotor.

Puntos operativos

$$\begin{aligned} \bar{\omega}_1 = \bar{\omega}_2 = \bar{\omega}_3 = \bar{\omega}_4 = \bar{\omega}_h, & \quad \bar{\omega}_h^2 = \frac{mg}{4k_F} \\ \bar{\phi} = \bar{\theta} = 0, & \quad \psi = \psi_0 \end{aligned} \quad (5)$$

Las ecuaciones no lineales se linealizaron alrededor de los puntos de funcionamiento y se dan en (6) y (7).

Dinámica de traslación lineal

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} g(\theta c\psi_0 + \phi s\psi_0) \\ g(\theta s\psi_0 - \phi c\psi_0) \\ \frac{2k_F\omega_h}{m}(\omega_1 + \omega_2 + \omega_3 + \omega_4 - 4\omega_h) \end{bmatrix} \quad (6)$$

Dinámica de rotación lineal

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{2Lk_F\omega_h}{I_{xx}}(\omega_2 - \omega_4) \\ \frac{2Lk_F\omega_h}{I_{yy}}(\omega_3 - \omega_1) \\ \frac{2Lk_M\omega_h}{I_{zz}}(\omega_1 - \omega_2 + \omega_3 - \omega_4) \end{bmatrix} \quad (7)$$

Las entradas se dan en los términos $\Delta\omega_F, \Delta\omega_\phi, \Delta\omega_\theta$ y $\Delta\omega_\psi$, causan movimiento vertical, roll, paso y yaw, respectivamente. Su relación con las velocidades individuales del rotor se muestra en (8). Las ecuaciones linealizadas se escribieron entonces en términos de las cuatro entradas $\Delta\omega_F, \Delta\omega_\phi, \Delta\omega_\theta$ y $\Delta\omega_\psi$, y, y se muestran en (9).

Se observa que $\ddot{z}, \dot{p}, \dot{q}$ y \dot{r} son controlados directamente por las entradas, \ddot{x} y \ddot{y} son afectados por los ángulos ϕ y θ . Para los pequeños ϕ y θ , $\dot{\phi} \approx p$ y $\dot{\theta} \approx q$.

El modelo se linealiza y se expresa en forma de espacio de estado. Las velocidades del rotor se utilizan para definir entradas y por lo tanto se utilizarán para el control. La linealización se llevó a cabo utilizando la aproximación de la serie de Taylor de primer orden [9-14]. Los puntos de operación son $z=3m$ y en $x=2m$, obteniendo las respuestas que se observan en la Figura 4. En donde los puntos de operación son alcanzados en un tiempo máximo de $t=10$ seg.



Relación de entradas y velocidades del rotor

$$\begin{aligned} \Delta\omega_F + \omega_h &= \frac{1}{4}(\omega_1 + \omega_2 + \omega_3 + \omega_4) \\ \Delta\omega_\phi &= \frac{1}{2}(\omega_2 - \omega_4) \\ \Delta\omega_\theta &= \frac{1}{2}(\omega_3 - \omega_1) \\ \Delta\omega_\psi &= \frac{1}{4}(\omega_1 - \omega_2 + \omega_3 - \omega_4) \end{aligned} \tag{8}$$

Dinámica con entradas redefinidas

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} g(\theta c\psi_0 + \phi s\psi_0) \\ g(\theta s\psi_0 - \phi c\psi_0) \\ \frac{2k_F\omega_h}{m}\Delta\omega_F \end{bmatrix}; \quad \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{4Lk_F\omega_h}{I_{xx}}\Delta\omega_\phi \\ \frac{4Lk_F\omega_h}{I_{yy}}\Delta\omega_\theta \\ \frac{8Lk_M\omega_h}{I_{zz}}\Delta\omega_\psi \end{bmatrix} \tag{9}$$

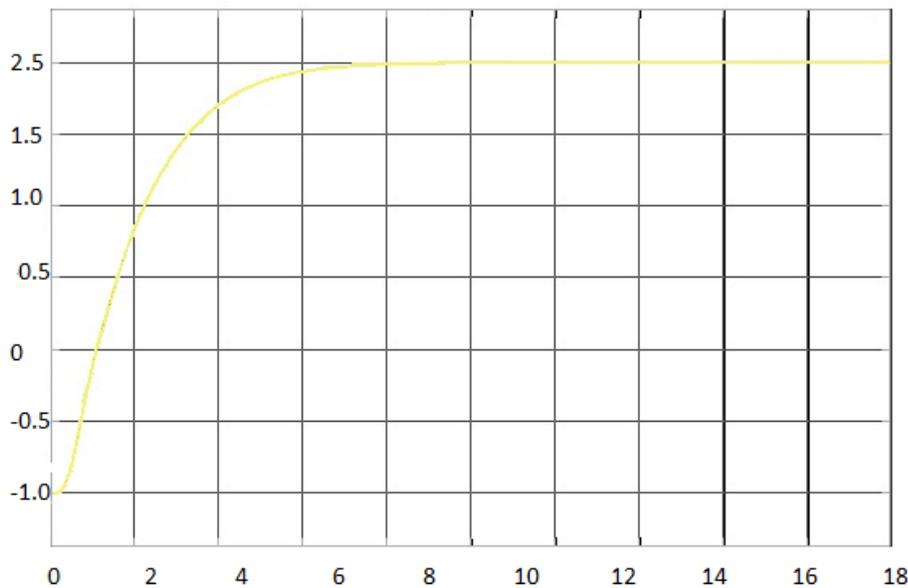


Figura 4. Respuesta en el tiempo del cuadricóptero Ar. Drone 2.0.

El cuadricóptero puede llevar a cabo cuatro movimientos diferentes independientemente: el movimiento vertical puede lograrse aumentando la potencia de todos los rotores simultáneamente, se puede lograr el balanceo y el paso aumentando y disminuyendo las velocidades de los rotores opuestos y la guiñada causando una diferencia en velocidad y por lo tanto par de rotores girando direcciones opuestas. Estos cuatro movimientos se producen debido a sumas y diferencias en las velocidades del rotor y están completamente desacoplados entre sí. Esto lleva a que las cuatro



entradas se definan como sumas y diferencias de velocidades del rotor en lugar de las velocidades de rotor individuales [20-22]. Con ayuda de MatLab [23] se obtiene la simulación con los puntos de operación referenciados en tiempo real del cuadricóptero AR Drone 2.0 en tiempo real, ver Figura 5.

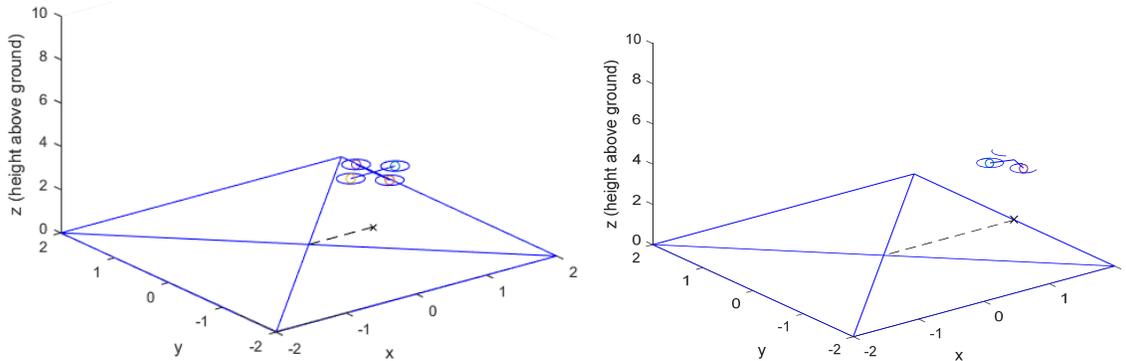


Figura 5. Simulación en el tiempo real del cuadricóptero Ar. Drone 2.0.

3.2 Arquitectura de control

Los controladores PID fueron seleccionados por las siguientes razones:

- 1) Los reguladores PID se utilizan ampliamente en la literatura [15] [11] [14] con resultados positivos;
- 2) Son intuitivos, simples de sintonizar y robustos a algunos ruidos y perturbaciones [16]; y
- 3) Los controladores de orden superior son computacionalmente más caros, aumentando el tiempo de procesamiento de los comandos de control.

Los controladores PID no se utilizaron para el control de posición debido a problemas de bobinado integral y sobreimpulso. El control integral se implementa generalmente para eliminar el error de estado estacionario, sin embargo, puesto que la posición es la segunda integral de aceleración que es proporcional a la variable controlada, la actitud, el error de estado estacionario será cero y no se requiere un método anti-windup [19-20].

4. Resultados

El uso del SDK para realización de proyectos puede resultar un tanto engorroso para algunos usuarios que tengan escaso o nulo conocimientos de programación o sistemas operativos, ya que este modelo fue especialmente diseñado para una manipulación en Linux en esta sistema se encuentran un sin número de proyectos dado que el AR Drone 2.0 cuenta con open source y es de fácil acceso; sin embargo, no tan amigable. En la Figura 6 se presenta el Cuadricóptero AR Drone 2.0 que es usado para adecuar el algoritmo de control de las trayectorias especificadas.

Las técnicas que se basan en controlar la posición combinando varias medidas con conocimiento previo de los sistemas de medida. La principal ventaja es que la estimación de control en el cuadricóptero AR. Drone 2.0 sea más precisa y fiable. El principal inconveniente es que pueden requerir linealizar en un punto de operación para controlar en posición y velocidad. Además, estas técnicas requieren un desarrollo matemático basado en los modelos de espacio de estado. El control básico en el cuadricóptero AR. Drone 2.0 puede ser segmentada en dos fases, de actualización de la percepción y la acción de control y utilizar en la integración de tecnologías como mediciones de temperatura, ver Figura 7, y medición de alturas, ver figura 8.



Figura 6. Cuadricóptero AR Drone 2.0.

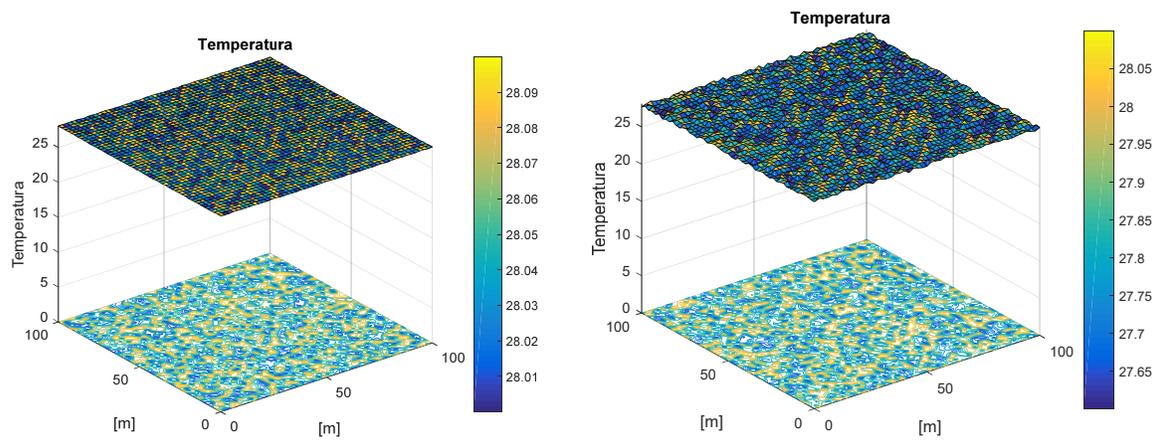


Figura 7. Medición de temperatura durante el vuelo del cuadricóptero AR Drone 2.0

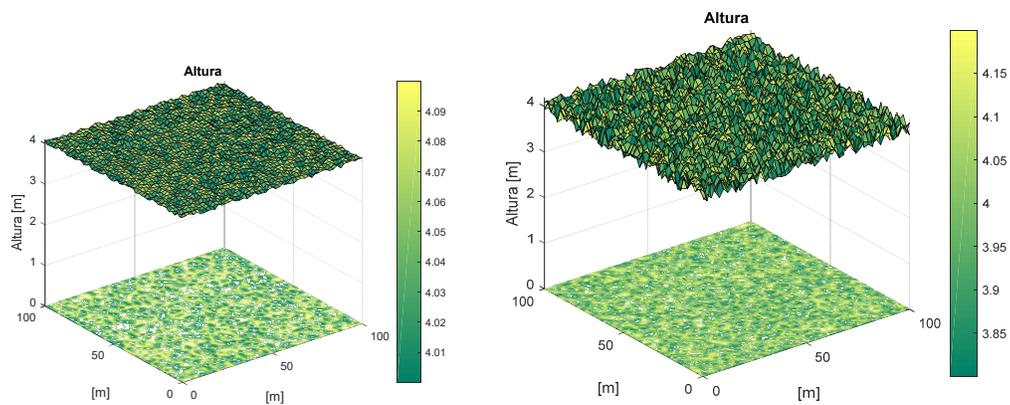


Figura 8. Medición de altura durante el vuelo del cuadricóptero AR Drone 2.0



5. Conclusiones

El algoritmo de control para el cuadricóptero AR Drone 2.0 tiene el objetivo de mejorar los atributos como la posición, velocidad y precisión del mismo. El control cinemático selecciona trayectorias que el robot deberá de seguir para que, teniendo en cuenta sus limitaciones ajustarse a los valores deseados por el usuario y con esto se puede aprovechar para integrar tecnologías en el control de navegación del Cuadricóptero AR Drone 2.0.

Referencias

- [1] Y. Sun, (2012). Modeling, Identification and Control of a Quad-rotor drone using low-resolution sensing. Master of Science in Mechanical Engineering in the Graduate College of the University of Illinois at Urbana-Champaign.
- [2] N. Dijkshoorn, (2012). Simultaneous localization and mapping with the AR.Drone. Masters Thesis for the graduation in Artificial Intelligence, Universiteit van Amsterdam.
- [3] M. Mogenson, (2012). The AR Drone LabVIEW Toolkit: A Software Framework for the Control of Low-Cost Quadrotor Aerial Robots, Master's Thesis in Mechanical Engineering, Tufts University.
- [4] P. Bouffard, (2012). On-board Model Predictive Control of a Quadrotor Helicopter: Design, Implementation, and Experiments, Technical Report No. UCB/EECS-2012-241 [5] E. Xargay, V Dobrokhodov, I. Kaminer, et al. Time Critical Cooperative Control of Multiple Autonomous Vehicles (2012), IEEE Control Systems Magazine, October 2012.
- [5] S. Piskorski, N. Brulez, P. Eline, F. D'Haeyer, (2012). AR.Drone Developer Guide, SDK 2.0 [7] P.-J. Bristeau, F. Callou, D. Vissiere, N. Petit, (2011). The Navigation and Control technology inside the AR.Drone micro UAV Preprints of the 18th IFAC World Congress Milano August 28 - September 2, 2011, pp. 1477-1484.
- [6] N. Shimpuku (puku0x), (2013). CV Drone Free Software on <https://github.com/puku0x/cvdrone>.
- [7] J. J. Engel, (2011). Autonomous Camera-Based Navigation of a Quadrocopter. Masters Thesis in Informatik, Technischen Universitat Munchen.
- [8] Hernandez, Andres, et al. "Identification and path following control of an AR. Drone quadrotor." System Theory, Control and Computing (ICSTCC), 2013 17th International Conference. IEEE, 2013.
- [9] Pleban, Johann-Sebastian, Ricardo Band, and Reiner Creutzburg. "Hacking and securing the AR. Drone 2.0 quadcopter: investigations for improving the security of a toy." IS&T/SPIE Electronic Imaging. International Society for Optics and Photonics, 2014.
- [10] Lugo, Jacobo Jiménez, and Andreas Zell. "Framework for autonomous on-board navigation with the AR. Drone." Journal of Intelligent & Robotic Systems 73.1-4 (2014): 401-412.
- [11] Martin, Gerard Theodore. "Modelling and Control of the Parrot AR. Drone." The UNSW Canberra at ADFA Journal of Undergraduate Engineering Research 5.1 (2012).
- [12] LIU, Hao, et al. "Drone Detection Based on an Audio-Assisted Camera Array. En *Multimedia Big Data (BigMM)*", 2017 IEEE Third International Conference on. IEEE, 2017. p. 402-406.
- [13] Carla Martínez Poveda, "Aplicaciones sobre AR Drone". Escola Tècnica Superior d'Enginyeria Informàtica Universitat Politècnica de València
- [14] PISKORSKI, Stephane, et al. "Ar. drone developer guide. Parrot, sdk", 2012, vol. 1.v
- [15] P. Corke., "System Identification-Theory for the User Robotics, Vision and Control", Springer Tracts in Advanced Robotics, Springer-Verlag, 2013.
- [16] R. Siegwart, I. Nourbakhsh, "Introduction to Autonomous Mobile Robots", The MIT Press, 2004.
- [17] S.K. Rao, T. Kailath, "Orthogonal Digital Filters for VLSI Implementation", IEEE Trans. Circuits Syst., CAS-31, 1984, pp. 933-945. Siciliano, B., Khatib, O., Springer Handbook of Robotics, Springer, 2008.
- [18] Latombe, J.C. (1991). "Robot motion Planning". Kluwer Academic Publishers.
- [19] Laumont, J.P., P.E. Jacobs, M. Taix and M. Murray (1994). "A Motion Planner for Nonholonomic Mobile Robots". IEEE Trans. on Robotics and Autom, Vol 10, No 5: 577-593.
- [20] Wolfram Burgard, Cyrill Stachniss, Kai Arras, Maren Bennewitz, "Introduction to Mobile Robotics SLAM: Simultaneous Localization and Mapping".



- [21] Ting-Yung Wen, J. (1997), "Control of Nonholonomic System". The Control Handbook, CRC Press, IEEE Press.
- [22] Murray, R., y S. Sastry (1990). "Steering Nonholonomic System Using Sinusoids". Proc. of the IEEE Int. Conf. on Decision and Control, pp. 1136-1141.
- [23] Robotics, Vision & Control: (c) Peter Corke 1992-2011 <http://www.petercorke.com> - Robotics Toolbox for Matlab (release 9.10) - pHRIWARE (release 1.1): pHRIWARE is Copyrighted by Bryan Moutrie (2013-2017) (c). Run rtbdemo to explore the toolbox