Congreso Nacional de Control Automático 2013
Ensenada, Baja California, Mexico, Octubre 16-18, 2013

Implementación en Tiempo Real Basada en el Modelo de Giotto de un Control no Lineal para Seguimiento de Trayectorias en un Cuadrirotor.

J. Rogelio Guadarrama-Olvera, Hugo Rodríguez-Cortés, Rafael Castro-Linares
Centro de Investigación y Estudios Avanzados del IPN, Av. Instituto Politécnico Nacional No 2508, Col. San Pedro Zacatenco 07360 México, DF
(jguadarrama,hrodriguez,rcastro)@cinvestav.mx

***Resumen*— Este artículo presenta una metodología detallada para la implementación en tiempo real del controlador no lineal propuesto en (Corona-Sánchez and Rodríguez-Cortés, 2013). La implementación en tiempo real se lleva a cabo utilizando el modelo de *Giotto* (Henzinger, Horowitz and Kirsch, 2001) para describir el cómputo llevado a cabo por el controlador. Además se agregan algunas modificaciones al controlador para darle robustez ante incertidumbres en los parámetros empleados en un helicóptero de cuatro rotores.**

Palabras clave: Sistemas en tiempo real, Cuadrirotor, Control no lineal, Giotto.

I. Introducción

Los avances tecnológicos de las últimas décadas en los diversos campos de la ingeniería han abierto las puertas a la implementación de sistemas cada vez más integrados, y por consiguiente, complejos en arquitectura. Gracias a la miniaturización de la electrónica, el desarrollo de nuevos sensores, actuadores y materiales compuestos, se cuenta ya con herramientas útiles para probar nuevas estrategias de control no lineales que pueden incluir funciones complejas en el lazo de retroalimentación o discontinuidades. Al mismo tiempo aparece una nueva gama de problemas de implementación pues al tener multiples dispositivos trabajando en conjunto, se requiere una planificación cuidadosa de los procesos para evitar bloqueos, pérdidas de sincronización o comportamiento diferente al planeado.

Para validar las soluciones a estos nuevos retos de enfoque tecnológico, se han utilizado diferentes plataformas experimentales, entre las cuales destacan las aeronaves no tripuladas o UAV tanto de ala fija como rotativa. Siendo los cuadrirotores o hélicopteros de cuatro rotores la plataforma experimental mas común debido a que son sistemas que no presentan complejidades mecánicas; sin embargo al ser subactuados requieren de sistemas de control en tiempo real relativamente complejos.

El problema de vuelo controlado para el cuadrirotor ha sido resuelto mediante diversos controladores que van desde lineales proporcional-derivativo hasta complejas estrategias no lineales logrando desde la simple estabilización en vuelo suspendido hasta la realización de maniobras de vuelo agresivas (Mellinger, Michael and Kumar, 2012).

El modelo dinámico del cuadrirotor ha sido ampliamente abordado por la comunidad científica. Los trabajos de (Castillo, Albertos, Garcia-Gil and Lozano, 2007) y (Das, Lewis and Subbarao, 2009) detallan el modelado del sistema mediante las ecuaciones de Euler-Lagrange y proponen un controlador basado en la técnica backstepping para estabilizar el helicóptero en vuelo estacionario. El controlador es probado en simulaciones y en una plataforma experimental en tiempo real en (Castillo et al., 2007) sin abundar en los detalles de la implementación, lo cual es una práctica común en la literatura sobre el Cuadrirotor.

En los trabajos de (Kendoul, Lara, Fantoni and Lozano, 2007) y (Santos, Romero, Salazar and Lozano, 2013) se presentan implementaciones en tiempo real de controladores no lineales sin especificar la estrategia o planificador utilizado para lograr que el sistema trabaje en tiempo real, mostrando resultados experimentales donde se aprecia la estabilización de la aeronave. Otra práctica común en la implementación en tiempo real de controladores para el cuadrirotor es la utilización de dos procesadores, uno para el sensado y control de la posición, que generalmente esta en tierra, y otro a bordo de la aeronave para la medición y control de la orientación, (Santos et al., 2013), (Hehn and D'Andrea, 2011), (Dydek, Annaswamy and Lavretsky, n.d.) y (Mellinger et al., 2012), ambos comunicándose por radiofrecuencia o Wi-Fi. También es frecuente la delegación de tareas periódicas a microcontroladores, encargándoles las tareas de más bajo nivel para no saturar al procesador principal con el manejo del hardware (Grzonka, Grisetti and Burgard, 2012).

Una práctica frecuente al proponer estrategias de control basadas en el modelo es la idealización del hardware como el hecho de suponer que todos los motores son exactamente idénticos; esto raras veces es cierto en plataformas experimentales debido a las tolerancias de fabricación y ensamble. Este punto es abordado por (Khebbache, Sait, Yacef and Soukkou, 2012) donde proponen una ley de control basada en la técnica Backstepping a la cual se agregan componentes saturados a la acción de control con los cuales se compensa la pérdida de potencia en los actuadores, se presentan resultados de simulación pero no de implementación en un

sistema físico.

En este artículo se implementará una ley de control no lineal utilizando sensores que operan a tazas de muestreo diferentes para el seguimiento de trayectorias en un cuadrirotor. A partir del algoritmo de *Giotto*, se presenta una descripción detallada de como las tareas de control se expresan como tareas de tiempo real utilizando un solo procesador. Dicho procesador también se encarga de la operación de sensores y actuadores, es decir, opera todos los niveles de control tomando la información e un sistema de posicionamiento global basado en cámaras infrarrojas y una central de medición inercial. La propuesta de implementación se valida experimentalmente.

Este artículo esta organizado de la forma siguiente. En la sección II se presenta el control a implementar (Corona-Sánchez and Rodríguez-Cortés, 2013), adicionando algunos ajustes para darle robustez al control de posición. En la sección III se presenta el Modelo de Giotto (Henzinger et al., 2001) mediante el cual se describe y propone la ejecución de las tareas. En la sección IV se detalla la implementación del controlador en la plataforma experimental. La sección V presenta los resultados experimentales y finalmente se presentan algunas conclusiones en la sección VI.

II. Estrategia de control.

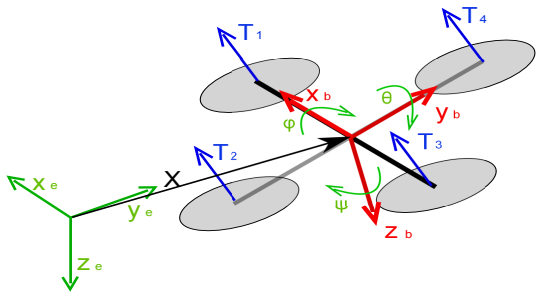

Figura 1. Modelado del cuadrirotor.

En este trabajo, el modelo dinámico que se considera es el que resulta de las ecuaciones de quasi-Euler-Lagrange con la variante de utilizar un marco de referencia cuerpo no inercial para describir las variables generalizadas (Meirovitch, 1991). El marco inercial se define con la convención Norte-Este-Abajo y el marco de referencia cuerpo se escoge como se muestra en la Figua 1. Para describir la orientación entre el marco de referencia inercial y el marco de referencia cuerpo se utiliza la secuencia de rotación ZYX de los ángulos de Euler, teniendo así que la cinemática traslacional y la dinámica traslacional queda descrita por las ecuaciones

$$\dot{X} = R\left(\Phi\right)^{T} V^{b} \qquad \Phi = \left[\begin{array}{ccc} \phi & \theta & \psi \end{array}\right]\top \tag{1}$$

$$R(\Phi) = \left[\begin{array}{ccc} c\theta c\psi & c\theta s\psi & -s\theta \\ c\psi s\theta s\phi - c\phi s\psi & c\phi c\psi + s\theta s\phi s\psi & c\theta s\phi \\ s\phi s\psi + c\phi c\psi s\theta & c\phi s\theta s\psi - c\psi s\phi & c\theta c\phi \end{array}\right] \tag{2}$$

$$m\dot{V}^{b} + \Omega \times mV^{b} - R\left(\Phi\right) mge_{1} = e_{1}T_{T} \tag{3}$$

dónde $\dot{X}$ es el vector de velocidades del centro de masa con respecto al marco de referencia inercial, V^b es el vector de velocidad traslacional del centro de masa referenciados al marco de referencia cuerpo, Φ es la orientación representada por los ángulos de euler (ϕ es el alabeo, θ es el cabeceo y ψ es la guiñada) y $R(\Phi)$ es la matriz de rotación[1], m y g son la masa y la aceleración gravitacional respectivamente y T_T es el empuje total generado por los motores y $e_1^b = \left[\begin{array}{ccc} 0 & 0 & 1 \end{array}\right]\top$. Así también se define la dinámica de la orientación expresada por.

$$\dot{\Phi} = W\left(\Phi\right)^{-1}\Omega \qquad W = \left[\begin{array}{ccc} 1 & 0 & -s\theta \\ 0 & c\phi & c\theta s\phi \\ 0 & -s\phi & c\theta c\phi \end{array}\right] \tag{4}$$

$$J\dot{\Omega} + \Omega \times J\Omega = M^{b} \tag{5}$$

dónde J es la matriz de inercia, W es una transformación para pasar las velocidades angulares del marco de referencia corporal al inercial, Ω es el vector de velocidad angular del centro de masa expresado en el marco de referencia cuerpo y M^b es el vector de momentos generados por diferencias entre los empujes de los motores.

El control que se implementará es el propuesto en (Corona-Sánchez and Rodríguez-Cortés, 2013) que plantea un control en tres niveles: uno para la posición en el espacio dividido en dos bloques, control de altura y control en el plano horizontal; en el segundo nivel un control para la orientación que recibe las referencias de ángulos deseados desde el nivel anterior y, por último, el control de bajo nivel para los actuadores.

II-A. Control de altura.

La dinámica vertical en coordenadas $z_1 = z - z_d$, $z_2 = \dot{z} - \dot{z}_d$ y $z_3 = \int_0^t z_1(\tau)d\tau$ esta descrita por las ecuaciones siguientes

$$\begin{array}{rcl} \dot{z}_1 & = & z_2 \\ \dot{z}_2 & = & -c_\theta c_\phi \frac{T_T}{m} + g - \ddot{z}_d \\ \dot{z}_3 & = & z_1 \end{array}$$

definiendo a la entrada de control T_T como

$$\gamma_z = -\ddot{z_d} + \frac{\epsilon_z}{2}\left[\tanh\left(\frac{2\lambda_{z3}}{\epsilon_z}z_3\right) + \frac{1}{2}\tanh\left(\frac{4\lambda_{z2}}{\epsilon_z}z_2\right) + \frac{1}{4}\tanh\left(\frac{8\lambda_{z1}}{\epsilon_z}z_1\right)\right] \tag{6}$$

De tal forma que la dinámica en lazo cerrado es

$$\begin{array}{rcl} \dot{z}_1 & = & z_2 \\ \dot{z}_2 & = & \gamma_z - \ddot{z}_d \\ \dot{z}_3 & = & z_1 \end{array}$$

dónde los parámetros ϵ_z, λ_{z1}, λ_{z2} y λ_{z3} son las ganancias a sintonizar. Al implementarse en la plataforma experimental, el control para la dinámica vertical propuesto en (**?**) se

[1] $s\theta = sin(\theta)$. $c\theta = cos(\theta)$ para simplificar la notación.

mostró vulnerable a la pérdida de potencia en los actuadores debida a la descarga no monitoreada de la batería. Esto produjo un error en estado estacionario, el cual fue eliminado agregando la acción integral.

Para sintonizar este control, se realiza una aproximación lineal para poder asignar polos con parte real negativa. La linealización resulta como sigue.

$$\begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \\ \dot{z}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -\lambda_1 & -\lambda_2 & -\lambda_3 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} \tag{7}$$

se puede verificar que con la selección adecuada de las ganancias λ_{zi}, $i = 1, 2, 3$ los polos del sistema linealizado (7) se pueden ubicar en el lugar deseado.

II-B. Control en el plano X-Y.

El control en el plano se realiza definiendo referencias adecuadas para la orientacion del vehículo. Dichas referencias, son

$$\Phi_d = \begin{bmatrix} \arctan\left(\frac{c_\theta(s_\psi\gamma_x - c_\psi\gamma_y)}{g-\gamma_z}\right) & \arctan\left(\frac{c_\psi\gamma_x + s_\psi\gamma_y}{g-\gamma_z}\right) & \psi_d \end{bmatrix}^\top \tag{8}$$

donde las funciones γ_x y γ_y (Corona-Sánchez and Rodríguez-Cortés, 2013) dependen del error de posición y velocidad. Estas funciones también están acotadas mediante la tangente hiperbólica y al igual que en el control de la dinámica vertical, son vulnerables a las diferencias de los empujes generados por los motores, produciendo un error de estado estacionario en la posición en el plano.

Este error también fue compensado mediante la adición de una acción integral a la ley de control, creando un par de estados mas correspondientes a las integrales del error de posición en x e y. Por lo tanto las nuevas funciones γ_x y γ_y quedan como

$$\gamma_k = -\ddot{k_d} - \frac{\epsilon_k}{2}\left[\tanh\left(\frac{2\lambda_{k3}}{\epsilon_k}k_3\right) + \frac{1}{2}\tanh\left(\frac{4\lambda_{k2}}{\epsilon_k}k_2\right) + \frac{1}{4}\tanh\left(\frac{8\lambda_{k1}}{\epsilon_k}k_1\right)\right] \quad \text{con} \quad k = x, y \tag{9}$$

Esta vez, los parámetros λ_{k1} ajustan la acción proporcional, λ_{k2} la acción derivativa y λ_{k3} la acción integral. Los parámetros ϵ_k ajustan los límites para la saturación de los controles virtuales, de manera que si se limitan las aceleraciones deseadas, se limitan los ángulos de orientación deseados.

II-C. Control de orientación

El controlador propuesto utiliza la técnica $Backstepping$ para interconectar el control de posición con el control de orientación mediante las acciones virtuales de control representadas por la orientación deseada. Posteriormente se estabiliza la orientación con un control basado en pasividad, específicamente basado en el moldeo de la interconexión y el amortiguamiento. A esta etapa del controlador no se le realizó ningún ajuste para la implementación.

Para evitar singularidades en la definición de los ángulos deseados la constante ϵ_z se selecciona de manera que la siguiente condición se cumpla

$$g > |\ddot{z}_d| + \frac{7\epsilon_z}{8} \tag{10}$$

es evidente que la aceleración deseada debe también acotarse.

II-D. Colocación del control.

Una vez calculados los pares necesarios M^b y el empuje T_T se calculan los empujes necesarios para cada motor mediante la transformación lineal.

$$\begin{bmatrix} T_T \\ M^b \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & l & 0 & -l \\ l & 0 & -l & 0 \\ -\frac{C_Q}{C_T}r & \frac{C_Q}{C_T}r & -\frac{C_Q}{C_T}r & \frac{C_Q}{C_T}r \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \end{bmatrix} \tag{11}$$

dónde la matriz está formada por los parámetros físicos (longitud del brazo l y radio de la hélice r) y aerodinámicos de la aeronave (Coeficiente de empuje C_T y de momento de reacción C_Q, (Leishman, 2006)) y T_i son los empujes de cada motor. Los coeficientes C_T y C_Q fueron aproximados experimentalmente.

III. El modelo de Giotto.

Para implementar efectivamente el controlador no lineal en la plataforma física se requiere poner a trabajar un conjunto de dispositivos electrónicos de medición, procesamiento y potencia con restricciones diferentes que pueden entrar en conflicto al ser operados a diferentes frecuencias de muestreo.

Para evitar esta clase de problemas se requiere que la planificación de las tareas de cómputo atienda todas las necesidades de los dispositivos sin perder la sincronía con ninguno de ellos y sin desatender las tareas con restricciones de tiempo menores.

El modelo de $Giotto$ (Henzinger et al., 2001) plantea un algoritmo para implementar sistemas donde se tienen tareas activadas por requerimientos de hardware a diferentes periodos, a la par de tareas de menor restricción temporal pero también indispensables. El modelo en cuestión plantea la transición entre la ley de control y la implementación en tiempo real del sistema.

Para poder implementar satisfactoriamente el modelo de $Giotto$ se debe conocer las restricciones temporales, tiempo de ejecución y precedencia de las tareas a programar para poder armar un modelo $Giotto$ del sistema. Para proponer dicho modelo se deben tomar en cuenta algunas definiciones.

Puerto: Son las entradas y salidas de información de las tareas, las cuales pueden estar conectadas a algún periférico (sensor o actuador) o entre tareas. Tienen la característica de mantener su valor constante hasta ser actualizados por algún evento.

Tarea: Son programas que toman la información del puerto de entrada, la procesan y entregan el resultado en su puerto de salida. La ejecución de las tareas puede ser invocada por restricción temporal o por precedencia. Los puertos no pueden ser modificados durante la ejecución de ésta.

Invocación: Son el conjunto de eventos que llaman a una tarea a ejecutarse, pueden ser el cumplimiento de un periodo de tiempo, alguna interrupción externa o la conclusión de

tareas previas cuyos puertos de salida conforman el puerto de entrada de la tarea invocada.

Modo: Se llama así a un acomodo de las tareas interconectando sus puertos que puede trabajar periódicamente por un tiempo indefinido y sólo puede cambiar cuando algún evento dispare un cambio de modo.

Cambio de modo: Similar a la invocación de las tareas, el cambio de modo es un evento que modifica el acomodo de las tareas pudiendo activar o desactivar la ejecución de algunas. Puede ser detonado por el cumplimiento de algun periodo de tiempo, un conteo de ciclos del modo anterior o algún evento externo. El cambio de modo debe respetar la ejecución de la última tarea del modo anterior.

IV. Implementación en tiempo real.

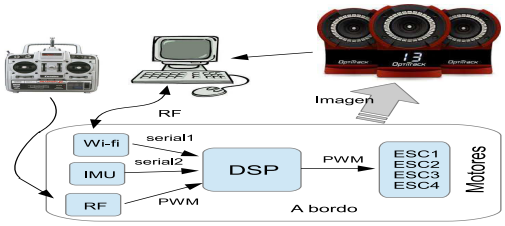

Figura 2. Arquitectura del sistema en tiempo real.

El Hardware que se va a utilizar en el cuadrirotor es el siguiente.

Central de medición inercial (IMU): Mide los vectores Ω y Φ, opera automáticamente enviando los datos en formato flotante a una frecuencia de 120 Hz, a través de una comunicación serial a 115200 baudios.

Sistema de visión Optitrack: Calcula en una computadora la posición del objeto en el espacio mediante la imagen capturada por un grupo de cámaras. Envía la medición por una señal de radio frecuencia con el protocolo TCP/IP Wi-Fi, ésto a una frecuencia de 60 Hz.

Módulo Wi-Fi: A bordo del helicóptero, recibe la señal de radiofrecuencia y la convierte a una comunicación serial a 115200 baudios.

Sistema radio control: Compuesto por un radiocontrol RF a 79 MHz con su respectivo receptor a bordo del vehículo, que permite a un operador dar instrucciones para el cuadrirotor como despegue, aterrizaje o paro de emergencia.

DSP: Es el computador monoprocesador a bordo del vehículo que cuenta con los periféricos para comunicarse con el hardware. Cuenta con tres puertos seriales, modulos de captura para señales digitales, modulos de PWM para los motores y memoria suficiente para los procesos. El procesador trabaja a una velocidad de 150 MIPS[2].

Además de las restricciones temporales de los sistemas de medición, se tiene una restricción de hardware adicional. El $Buffer$ de entrada de los módulos de comunicación seriales son de 15 localidades de 8 bits y los mensajes entregados por los sensores exceden este tamaño por lo que deberán ser recibidos en bloques. Otro aspecto a considerar es que estos paquetes de datos deben ser interpretados para construir los valores flotantes, pues vienen en forma de arreglo de variables tipo $Byte$ y con ellos se constituye cada flotante. Por lo que la sincronía toma un papel primordial pues con un dato que se omita se pierde el paquete entero de datos y no puede ser interpretado. Por último, determinados los 3 niveles de control, la frecuencia de nivel de control está dada por el periferico que la alimenta, siendo los 120 Hz de la central inercial para el control de orientación y los 60 Hz del sistema de visión para el control de posición. Entonces se tienen dos procesos a diferentes frecuencias corriendo en un solo procesador.

Un último punto importante es que el control de velocidad de los motores se lleva a cabo por un ESC[3] comercial, el cual comúnmente recibe la información mediante una comunicación serial teniendo una actualización inmediata de la velocidad deseada. Existe una variante de ESC de un costo significativamente menor frecuentemente usada en helicópteros de modelismo, los cuales reciben las velocidades deseadas mediante una señal de PWM de frecuencia baja; esto presenta un problema adicional pues la actualización de velocidad deseada llega al motor con una latencia que en el peor de los casos es igual al periodo del PWM que lo opera. Los ESC utilizados en el Cuadrirotor trabajan con una señal de PWM a 400 Hz por lo que dicho periodo de retraso, en el peor de los casos es de 2.5 ms.

IV-A. Definición de las tareas.

Una vez consideradas todas las restricciones, se definen las tareas necesarias para el proceso.

T1, Lectura de IMU 1: Recepción del primer paquete de datos de la central inercial, invocada por la bandera de interrupción del $Buffer$ de entrada del módulo de comunicación serial 1. Tiempo de cómputo de 10 μs.

T2, Lectura de IMU 2: Recepción del segundo paquete de datos de la centra inercial, invocada por la bandera de interrupción del $Buffer$ de entrada del módulo de comunicación serial 1 y la precedencia de $T1$. Tiempo cómputo de 10 μs.

En $T1$ y $T2$ se reciben 6 datos de punto flotante de la IMU (ϕ, θ, ψ, p, q y r), que adicionando 4 bytes de encabezado de configuración y la suma para verificación forman un arreglo de 29 bytes. Tomando en cuenta que el $buffer$ de entrada del DSP sólo cuneta con 15 registros, es necesario partir dicho mensaje en 2 bloques, por eso esta acción se lleva a cabo en dos tareas.

T3, Lectura de Wi-Fi 1: Recepción del primer paquete de datos del módulo Wi-Fi, invocada por la bandera de interrupción del $Buffer$ de entrada del módulo de comunicación serial 2. Tiene un tiempo de cómputo de 10 μs.

T4, Lectura de Wi-Fi 2: Recepción del segundo paquete de datos del módulo Wi-Fi, invocada por la bandera de

[2]Millones de intrucciones por segundo

[3]Siglas del inglés *Electronic Speed Control*

interrupción del $Buffer$ de entrada del módulo de comunicación serial 2 y la precedencia de $T3$. Tiempo de cómputo de 10 μs.

En $T3$ y $T4$ se recibe la posición del cuadrirotor en forma de 3 datos de punto flotante (x, y y z), a los que se agregan 4 bytes para comandos (Despegue, aterrizaje, etc.) y la suma de verificación, formando un arreglo de 17 bytes que se dividen en 2 bloques para poder ser leidos por el DSP como en el caso de la orientación proporcionada por la IMU.

T5, Control de posición: Toma el arreglo recibido del módulo Wi-Fi, calcula los dátos en formato flotante, calcula la orientación deseada. Esta tarea es invocada por la precedencia de $T4$ con un tiempo de cómputo de 40 μs.

T6, Control de orientación: Toma el arreglo recibido de la central inercial y calcula los datos en formato flotante; posteriormente evalua la ley de control de orientación. A la salida entrega los valores de PWM necesarios para cada motor. Es invocada por la precedencia de $T3$ y tiene un tiempo de cómputo de 140 μs.

T7, Lectura RF: Recepción del estado del radio control, invocada por la bandera de interrupción del módulo de captura (ECAP), ocurre a una frecuencia de 50 Hz. A la salida entrega el valor recibido en formato flotante y su tiempo de cómputo es de 10μs.

T8, Sincronización IMU: Espera la recepción de la primer cadena válida de la central inercial y configura los registros para las interrupciones del $Buffer$ de entrada del módulo de comunicación serial 1 y todas las variables para la ley de control. Esta tarea es invocada por el encendido inicial. A la salida entrega una bandera lógica (r1: true). Su tiempo de cómputo es 8,56 ms.

T9, Sincronización Wi-Fi: Espera la recepción de la primer cadena válida del módulo Wi-Fi y configura los registros para las interrupciones del $Buffer$ de entrada del módulo de comunicación serial 2. Esta tarea es invocada por la precedencia de $T8$. A la salida entrega una bandera lógica (r2: true) y su tiempo de cómputo es 16,23 ms.

IV-B. Definición de los Modos.

Una vez definidas las tareas, se definen los modos de operación para el sistema como sigue.

- **M1, Sincronización:** Es el modo con el que arranca el sistema, compuesto por las tareas $T8$ y $T9$, su frecuencia principal esta indefinida.
- **M2, Operación:** Es el modo de operación principal del sistema, conformado por las tareas $T1$, $T2$, $T3$, $T4$, $T5$, $T6$ y $T7$ activado por el cumplimiento de un ciclo de $M1$; su frecuencia principal es de 120 Hz.

V. Resultados experimentales.

Una vez planteado el modelo de Giotto de la aplicación se procede a programarlo en el DSP, en forma concurrente protegiendo de interrupciones a las secciones donde se acceda a la memoria compartida que son al inicio y final de cada tarea. Esto de acuerdo al el modelo de Giotto, para que se mantenga el estado de los puertos entre eventos. La

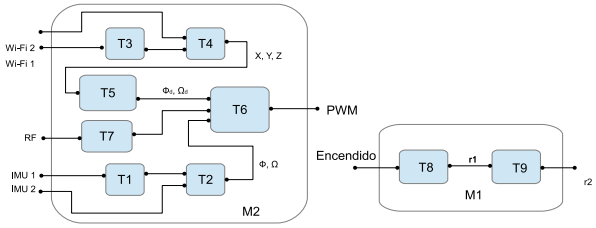

Figura 3. Modos de operación

ejecución de las tareas se lleva a cabo como se muestra en la Figura 4, tomando en cuenta que se tiene un solo procesador y las tareas con restricción de hardware tienen la más alta prioridad, es decir éstas pueden interrumpir a las tareas de menor prioridad. Dicha interrupción se puede apreciar en la Figura 4, donde la tarea $T6$ comienza en t_3 pero se ve interrumpida por $T4$ y $T7$ que tienen mayor prioridad , retomando su ejecución en t_6. También se puede ver cómo la tarea $T5$ es invocada en t_6 pero, al estar ocupado el procesador, no se ejecuta sino hasta que se termina la tarea 5 en t_7. En cuanto a las tareas $T1$, $T2$, $T3$ y $T4$, éstas pueden ser invocadas simultáneamente o mientras alguna de ellas está en ejecución, pero se llevarán acabo hasta que quede libre el procesador pues todas tienen la misma prioridad.

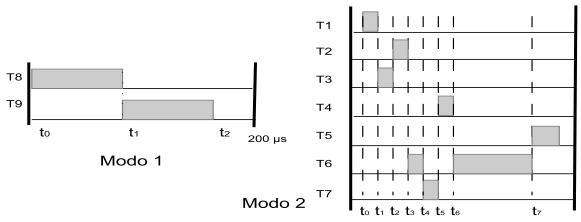

Figura 4. Uso del procesador en el modo 1.

A pesar de que la frecuencia base del modo es 120 Hz, el tiempo de cómputo total de las tareas es bastante menor y gracias al modelo de Giotto nunca se perderá la sincronía con los dispositivos. Al tiempo de cómputo total que es menor a 0.5 ms se adiciona el tiempo que tarda en llegar la actualización a los motores[4] que es de 2.5 ms, en el peor de los casos; sumados resulta un tiempo de 3 ms que es menor a los 8.33 ms del periodo de la tarea principal, lo cual asegura la planificabilidad del proceso.

La implementación se prueba en un cuadrirotor de 1 Kg en una trayectoria conformada por la elipse descrita por las ecuaciones siguientes

$$x_d = -\frac{1}{2} + \frac{3}{4}\sin\left(\frac{2\pi t}{20}\right)\text{m},\ y_d = \frac{1}{2}\cos\left(\frac{2\pi t}{20}\right)\text{m},\ z_d = -0{,}35\text{m}$$

La velocidad es de 3 ciclos por minuto. En la figura 5 se muestran las trayectorias en los ejes X, Y y Z del

[4]Este tiempo no se agregó en el gráfico de la figura 4 debido a que el retrazo ocurre a nivel de hardware y no ocupa al procesador.

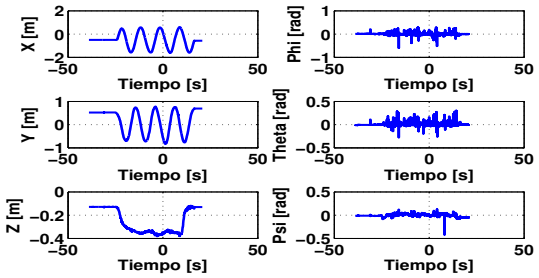

Figura 5. Resultados experimentales.

TABLA I
PARÁMETROS DEL QUADROTOR.

Parámetro	Valor	Parámetro	Valor
l	$0{,}22m$	C_T	0,014
m	$1{,}2Kg$	C_Q	0,004
r	0,127	I_{xx}	$0{,}1Kgm^2$
g	$9{,}81m/s2$	I_{yy}	$0{,}1Kgm^2$
ρ	$0{,}9Kg/m3$	I_{zz}	$0{,}17Kgm^2$

experimento. También muestra la orientación obtenida por el sistema optitrack a lo largo del experimento. Cabe señalar que el vehículo no utiliza la orientación medidad por el sistema Optitrack para calcular el control, sino la que recibe de la IMU, la cual no muestra el ruido de altas frecuencias con tanta amplitud. La figura 6 muestra la trayectoria recorrida por el cuadrirotor en perspectiva tridimensional.

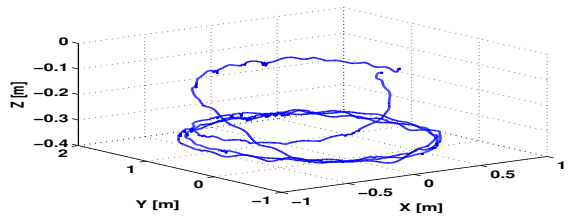

Figura 6. Trayectoria (13) en perspectiva.

VI. CONCLUSIÓN

Para sintetizar controladores basados en el modelo, se idealizan elementos que, en la práctica, presentan fallas debidas a tolerancias, ademas de fenómenos que no son modelados tales como los efectos de fricciones o de cambios térmicos que modifican ligeramente los parámetros del sistema, poniendo a prueba la robustez del sistema de control.

En este trabajo se realizaron algunos ajustes a la ley de control propuesta en (Corona-Sánchez and Rodríguez-Cortés, 2013) para dar robustez ante la pérdida de potencia en algunos motores, lo que causa un desbalance en la transformación lineal que calcula el empuje deseado para cada motor.

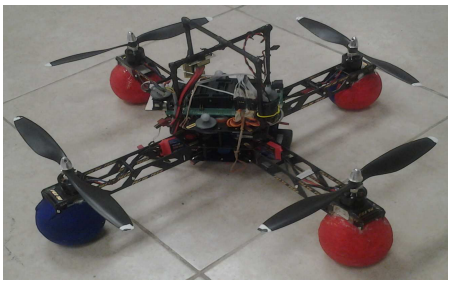

Figura 7. Plataforma experimental.

Además se presentó a detalle la metodología utilizada para dicha implementación utilizando el modelo de Giotto lo cual presenta una descripción más entendible y manejable del sistema haciendo más fácil la realización de posibles modificaciones futuras, como el cambiar el módulo Wi-Fi por un GPS para poder llevar el prototipo a un ambiente fuera del laboratorio.

La prueba experimental de la metodología resulto satisfactoria. Se eliminaron los errores de estado estacionario y se logró un seguimiento adecuado a la trayectoria sin la presencia de fallas en el sistema de tiempo real.

REFERENCIAS

Castillo, P., Albertos, P., Garcia-Gil, P. and Lozano, R. (2007), 'Modelado y estabilización de un helicóptero con cuatro motores', *Revista Iberoamericana de Automática e Informática Industrial* **4**(1), 41–57.

Corona-Sánchez, J. J. and Rodríguez-Cortés, H. (2013), 'Trajectory tracking control for a rotary wing vehicle powered by four rotors', *Journal of Intelligent & Robotic Systems* **70**(1-4), 39–50.

Das, A., Lewis, F. and Subbarao, K. (2009), 'Backstepping approach for controlling a quadrotor using lagrange form dynamics', *J. Intell. Robotics Syst.* **56**(1-2), 127–151.

Dydek, Z. T., Annaswamy, A. M. and Lavretsky, E. (n.d.), 'Adaptive control of quadrotor uavs: A design trade study with flight evaluations'.

Grzonka, S., Grisetti, G. and Burgard, W. (2012), 'A fully autonomous indoor quadrotor', *Robotics, IEEE Transactions on* **28**(1), 90–100.

Hehn, M. and D'Andrea, R. (2011), A flying inverted pendulum, *in* 'Robotics and Automation (ICRA), 2011 IEEE International Conference on', IEEE, pp. 763–770.

Henzinger, T. A., Horowitz, B. and Kirsch, C. M. (2001), Giotto: A time-triggered language for embedded programming, *in* 'Embedded Software', Springer, pp. 166–184.

Kendoul, F., Lara, D., Fantoni, I. and Lozano, R. (2007), 'Real-time nonlinear embedded control for an autonomous quadrotor helicopter', *Journal of guidance, control, and dynamics* **30**(4), 1049–1061.

Khebbache, H., Sait, B., Yacef, F. and Soukkou, Y. (2012), 'Robust stabilization of a quadrotor aerial vehicle in presence of actuator faults', *International Journal of Information Technology, Control and Automation* **2**(2), 1–13.

Leishman, J. G. (2006), *Principles of helicopter aerodynamics*, Cambridge University Press.

Meirovitch, L. (1991), 'Hybrid state equations of motion for flexible bodies in terms of quasi-coordinates', *Journal of Guidance, Control, and Dynamics* **14**(5), 1008–1013.

Mellinger, D., Michael, N. and Kumar, V. (2012), 'Trajectory generation and control for precise aggressive maneuvers with quadrotors', *Int. J. Rob. Res.* **31**(5), 664–674.

Pree, W., Templ, J., Hintenaus, P. and Naderlinger, A. (2011), 'Tdl - steps beyond giotto: A case for automated software construction', *International Journal of Software and Informatics* **5**(1-2), 335–354.

Santos, O., Romero, H., Salazar, S. and Lozano, R. (2013), 'Real-time stabilization of a quadrotor uav: Nonlinear optimal and suboptimal control', *Journal of Intelligent & Robotic Systems* **70**(1-4), 79–91.