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

Coordinación de movimiento para sistemas multiagente heterogéneos ⋆

M. A. Rosaldo Serrano * **E. Aranda Bricaire** *

* *Departamento de Ingeniería Eléctrica, Sección de Mecatrónica CINVESTAV, AP. 14-740, 07000 México DF, México. (e-mail: blitzchocobo@hotmail.com, earanda@cinvestav.mx)*

Resumen: En este artículo se presenta el problema de control de marcha para un sistema multiagente heterogéneo. El sistema esta conformado por un robot terrestre tipo uniciclo AmigoBot y un robot aéreo tipo quadrirotor AR Drone 2.0. Se describe el hardware y software de ambos robots, se obtiene el modelo cinemático del AmigoBot y el modelo dinámico del AR Drone. Finalmente se implementa una ley de control en cada robot para un esquema líder-seguidor.

Keywords: Sistemas Multi-Agente; Aeronaves no tripuladas; Robótica Móvil

1. INTRODUCCIÓN

Los sistemas multi-agente pueden ser definidos como conjuntos de robots autónomos coordinados a través de un sistema de comunicación para realizar tareas cooperativas. En la actualidad los sistemas multi-agente han encontrado un gran número de campos de aplicación tales como exploración terrestre y oceánica, ver Cao et al (2011). El uso de sistemas multi-agente ofrece ventajas al poder realizar tareas de mayor complejidad que las que un solo robot podría realizar. Además el sistema se vuelve mas flexible y tolerante a fallas, ver Yamaguchi (2003). El rango de las aplicaciones incluye cirugía a distancia con sistemas hapticos, ver Olfati-Saber et al (2002), el manejo de residuos tóxicos, transporte y manipulación de objetos de gran tamaño,ver Asahiro et al (1999), exploración, búsqueda y rescate, y la simulación del comportamiento de entidades biológicas, ver Arai et al (2002). La coordinación de movimiento es un área importante en la investigación de sistemas multi-agente, especialmente el control de formación, Chen et al (2005). El objetivo principal es coordinar un grupo de agentes para alcanzar una formación deseada. La estrategia de control es descentralizada por que se asume que cada agente conoce la posición de un subconjunto de agentes. El enfoque descentralizado ofrece mayor autonomía para los robots, menos carga de trabajo al implementar el control y es aplicable para grupos de gran escala, ver Do (2007). La gran mayoría de investigación de sistemas multi-agente esta enfocada a sistemas homogéneos, donde todos los agentes tienen el mismo modelo y poseen características de computo y sensado similares. En este artículo se aborda el caso de agentes heterogéneos, donde los modelos matemáticos de los agentes pueden ser diferentes entre ellos. Al considerar agentes heterogéneos se aumenta la cantidad de aplicaciones posibles pero también la complejidad del sistema de control. Existen artículos que abordan el problema de control de sistemas multi-agente heterogéneos, ver Chopra et al (2008). El problema de consenso para sistemas multiagente heterogéneos es presentado en Zheng et al (2011) mientras que Jönsson et al (2010) abordan una algoritmo de consenso aplicado a control de formación.

Para la realización de este artículo se utilizaron 2 tipos de robots autónomos diferentes. El robot tipo quadrirotor AR Drone 2.0 de Parrot y el robot tipo uniciclo Amigobot de Pioner. En la sección 2 se hace una descripción básica del hardware y software del AR Drone y se obtiene el modelo dinámico del mismo. En la sección 3 se hace una descripción básica del hardware y el software del AmigoBot y se obtiene el modelo cinemático del mismo. La sección 4 muestra el sistema de visión utilizado como retroalimentación. La sección 5 muestra el diseño de las leyes de control de los agentes y en la sección 6 se muestran simulaciones y experimentos realizados con dichas leyes de control.

2. DESCRIPCIÓN DEL AR DRONE 2.0

El AR Drone 2.0 es un vehículo aéreo no tripulado de tipo quadrirotor de uso civil recreativo fabricado por la compañía francesa Parrot. En esta sección se hace una breve introducción al AR Drone 2.0. Se describirá tanto el hardware como el software utilizado para controlarlo.

2.1 Características básicas

La estructura mecánica del drone se compone de cuatro motores unidos a las puntas de una estructura en forma de cruz que soporta la batería y los circuitos electrónicos de control. Para controlar el movimiento en el espacio del quadrirotor es necesario variar sus ángulos de cabeceo (inclinación respecto al eje Y_B del drone), alabeo (inclinación respecto al eje X_B del drone) y guiñada (rotación intrínseca alrededor del eje Z_B del drone). En la Fig. 1 se muestra el diagrama NED (North-West-Down) del AR Drone, Este tipo de diagrama es el mas comúnmente utilizado para vehículos aéreos.

⋆ Trabajo apoyado parcialmente por CONACYT, México, mediante la beca No. 345679

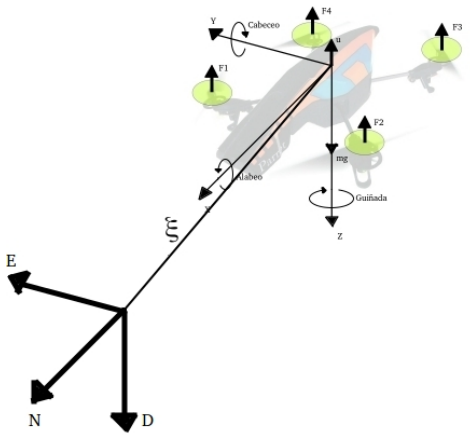

Figura 1. Diagrama NED del AR Drone 2.0

2.2 Hardware

El AR Drone 2.0 es un quadrirotor propulsado por motores eléctricos diseñado para juegos de realidad aumentada, como su nombre lo indica. En esta sección se describirán los componentes físicos con los que cuenta.

Estructura base La estructura base del AR Drone 2.0 consiste de una estructura principal de fibra de carbono en forma de cruz, un cuerpo de plástico y una capucha protectora. El drone incluye dos capuchas protectoras diferentes, una para exteriores compuesta de plástico ligero y una para interiores que incluye una protección de polipropileno expandido para las hélices. Ver Fig. 2.

Figura 2. Capuchas del AR Drone 2.0

Motores El AR Drone 2.0 cuenta con cuatro motores trifásicos sin escobillas controlados por corriente mediante un microcontrolador.

Baterias LiPo El drone es alimentado por una batería de polímero de Litio (Li-Po) de 11,1v con una capacidad de carga de 1000mAh con la cual puede realizar un vuelo de hasta 12 minutos, pudiendo alcanzar velocidades mayores a 5m/s.

Electrónica La placa de control del AR Drone cuenta con restricciones de seguridad tales como limite de altura, limite de inclinación para los ángulos de control de cabeceo y alabeo, limite de velocidad para el ascenso/descenso y ángulo de rotación, todas programables, además de ayudar al usuario con las secuencias de despegue y aterrizaje. El operador del drone puede seleccionar directamente sus ángulos de cabeceo y alabeo, la velocidad de rotación del ángulo de guiñada y la velocidad de ascenso/descenso. La placa de control del drone incluye una computadora basada en el procesador ARM9 con una velocidad de 468MHz y 128 MB de DDR RAM a 200MHz. Los sensores con los que cuenta el drone son una unidad inercial de medición (IMU) de 6 ejes conformada por un giroscopio y un acelerómetro de 3 ejes cada uno, un magnetómetro de 3 ejes, un sensor ultrasónico de altura, un sensor de presión y dos cámaras. La Fig. 3 muestra un diagrama de despiece del AR Drone 2.0.

Figura 3. Hardware del AR Drone 2

2.3 Software

La placa de control del drone tiene instalada la distribución de GNU/Linux BusyBox con el kernel 2.6.27. El software interno del drone provee la comunicación remota y se encarga de la estabilización del drone.
Para comunicar una computadora con el drone se utiliza el software de desarrollo proveído por el fabricante para el sistema operativo GNU/Linux Ubuntu. La interfaz utiliza tres canales de comunicación, cada uno con un puerto UDP (User Datagram Protocol) diferente, dichos puertos son el canal de comandos, el canal de datos de navegación (navdatas) y el canal de vídeo.

2.4 Modelo dinámico del AR Drone 2.0

En esta sección se presenta el modelo dinámico del quadrirotor obtenido mediante el método de Newton-Euler como se muestra en García et al (2013). Este modelo se obtiene representando al quadrirotor como un cuerpo rígido que se puede mover en un espacio tridimensional sujeto a una fuerza y a tres momentos. La dinámica de los cuatro motores eléctricos es relativamente rápida por lo que no se tomara en cuenta al igual que la flexibilidad de las hélices. Retomando el diagrama de la Fig. 1, representamos la posición del centro de masa del quadrirotor respecto al marco de referencia fijo mediante el vector $\xi = (x, y, z)^T$ y su orientación con respecto al marco inercial del quadrirotor mediante el vector $\eta = (\psi, \theta, \phi)^T$, donde ψ, θ y ϕ son los ángulos de Euler de guiñada, alabeo y cabeceo, respectivamente. La dinámica no lineal completa del quadrirotor puede ser expresada como:

$$m\ddot{\xi} = -mg\mathbf{D} + \mathbf{R}F \quad (1)$$

$$I\dot{\Omega} = -\Omega \times I\Omega + \tau \quad (2)$$

Donde $\mathbf{D}$ es un vector unitario sobre el eje D, $\mathbf{R}$ es una matriz de rotación que relaciona el marco de referencia fijo con el marco de referencia del drone, $\mathbf{F}$ representa la fuerza total aplicada al drone, m es su masa total, g es la constante de gravedad, Ω representa la velocidad angular del drone respecto al marco de referencia del mismo, I representa la matriz de inercia, y τ es el par total. Sea $u = \sum_{i=1}^{4} T_i$ la fuerza aplicada al drone,

la cual es generada por los cuatro motores. Asumiendo que esta fuerza tiene solamente una componente en la dirección de $-D$, la fuerza total puede ser escrita como $F = (0, 0, -u)^T$. Representando $\cos(*)$ y $\sin(*)$ como c_* y s_* respectivamente la matriz de rotación esta definida como

$$\mathbf{R} = \begin{bmatrix} c_\theta c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi & c_\phi s_\theta c_\psi + s_\phi s_\psi \\ c_\theta s_\psi & s_\phi s_\theta s_\psi + c_\phi c_\psi & c_\phi s_\theta s_\psi - s_\phi c_\psi \\ -s_\theta & s_\phi c_\theta & c_\phi c_\theta \end{bmatrix} \quad (3)$$

Definimos un vector auxiliar $\tilde{\tau}$ relacionado al par generalizado τ y basado en la ecuación 2:

$$\tilde{\tau} = \begin{bmatrix} \tilde{\tau}_\psi \\ \tilde{\tau}_\theta \\ \tilde{\tau}_\phi \end{bmatrix} = I^{-1}W^{-1}(-I\dot{W}\dot{\eta} - W\dot{\eta} \times IW\dot{\eta} + \tau) \quad (4)$$

Donde $\Omega = W\dot{\eta}$ y W es:

$$W = \begin{bmatrix} -\sin(\theta) & 0 & 1 \\ \cos(\theta)\sin(\phi) & \cos(\phi) & 0 \\ \cos(\theta)\cos(\phi) & -\sin(\phi) & 0 \end{bmatrix} \quad (5)$$

Utilizando las relaciones anteriores podemos representar el modelo dinámico del quadrirotor como

$$m\ddot{x} = -u(\cos(\psi)\sin(\theta)\cos(\phi) + \sin(\psi)\sin(\phi)) \quad (6)$$
$$m\ddot{y} = -u(\sin(\psi)\sin(\theta)\cos(\phi) - \cos(\psi)\sin(\phi)) \quad (7)$$
$$m\ddot{z} = -u(\cos(\theta)\cos(\phi)) + mg \quad (8)$$
$$\ddot{\psi} = \tilde{\tau}_\psi \quad (9)$$
$$\ddot{\theta} = \tilde{\tau}_\theta \quad (10)$$
$$\ddot{\phi} = \tilde{\tau}_\phi \quad (11)$$

Donde x, y son las coordenadas en el plano horizontal, z es la posición vertical, ψ es el ángulo de guiñada alrededor del eje z, θ es el ángulo de cabeceo alrededor del eje y (del quadrirotor), y ϕ es el ángulo de alabeo alrededor del eje-x(del quadrirotor). Las entradas de control $u, \tilde{\tau}_\psi, \tilde{\tau}_\theta$ y $\tilde{\tau}_\phi$ son el empuje total y los momentos angulares. Como se menciono con anterioridad el AR Drone 2.0 cuenta con un controlador interno que se encarga de posicionar al quadrirotor en cualquier ángulo solicitado por el usuario. Por ello podemos simplificar el modelo anterior utilizando solamente las ecuaciones (6)-(8), utilizando como entradas de control los ángulos de cabeceo, alabeo y guiñada. El movimiento del quadrirotor en el plano horizontal sera controlado mediante pequeñas variaciones en los ángulos θ y ϕ en un rango de -5^o a 5^o por lo que podemos considerar que $\sin(*) \approx *$ y que $cos(*) \approx 1$. Sustituyendo dichas consideraciones en las ecuaciones (6)-(8) obtenemos un nuevo modelo aún mas simplificado

$$m\ddot{x} = -u(\theta\cos(\psi) + \phi\sin(\psi)) \quad (12)$$
$$m\ddot{y} = -u(\theta\sin(\psi) - \phi\cos(\psi)) \quad (13)$$
$$m\ddot{z} = -u + mg \quad (14)$$

En este artículo se considera que el quadrirotor solamente se moverá en el plano horizontal y mantendrá una altura constante, con lo que de la ecuación (14) obtenemos que la fuerza de sustentación se equilibra con el peso del quadrirotor.

$$m(0) = -u + mg$$
$$u = mg \quad (15)$$

Sustituyendo (15) en (12) y (13) obtenemos un modelo mas simple para controlar al quadrirotor en el plano horizontal

$$m\ddot{x} = -(mg)(\theta\cos(\psi) + \phi\sin(\psi))$$
$$\ddot{x} = -g(\theta\cos(\psi) + \phi\sin(\psi)) \quad (16)$$
$$m\ddot{y} = -(mg)(\theta\sin(\psi) - \phi\cos(\psi))$$
$$\ddot{y} = -g(\theta\sin(\psi) - \phi\cos(\psi)) \quad (17)$$

Factorizamos las variables de control θ y ϕ para obtener el modelo dinámico simplificado del quadrirotor

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = -g \begin{bmatrix} \cos(\psi) & \sin(\psi) \\ \sin(\psi) & -\cos(\psi) \end{bmatrix} \begin{bmatrix} \theta \\ \phi \end{bmatrix} = A_d(\psi) \begin{bmatrix} \theta \\ \phi \end{bmatrix} \quad (18)$$

3. DESCRIPCIÓN DEL AMIGOBOT

El AmigoBot es un vehículo terrestre tipo uniciclo de la linea Pioneer de ActivMedia Robotics. En esta sección se realiza una breve introducción al AmigoBot. Se describirá tanto el hardware como el software utilizado para controlarlo.

3.1 Características básicas

AmigoBot es considerado un robot inteligente por que cuenta con un microcontrolador abordo cargado con software (AmigOS) y sonares ultrasónicos. Asimismo cuenta con Software dedicado para robótica como lo es la interfaz para aplicaciones en robótica de ActivMedia (ARIA por sus siglas en ingles).

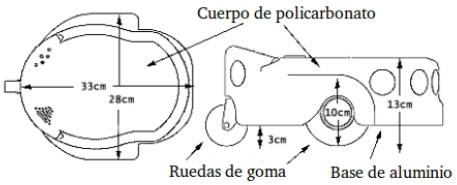

Figura 4. Características y dimensiones del AmigoBot

3.2 Hardware

AmigoBot esta diseñado para ser utilizado en interiores. Cuenta con ruedas solidas de goma de 4 pulgadas de diámetro, cada una conectada a un motor reversible de CD. La corriente de los motores es controlada por medio de PWM. Cada motor incluye un encoder de alta resolución que es utilizado por el microcontrolador del AmigoBot para determinar la velocidad de cada motor. Las dimen-

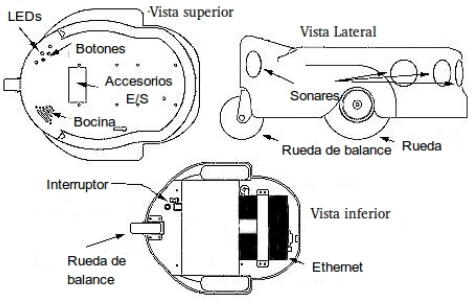

Figura 5. Características y dimensiones del AmigoBot

siones de un AmigoBot son 33cm de largo, 28cm de ancho y 13cm de alto. La longitud del eje de las ruedas 28cm, el

radio de las ruedas 5cm y tiene un peso de 3,6kg. Estos robots cuentan con un control PID interno para el control de velocidad lineal de cada rueda. Solo es necesario enviarles dos valores que corresponden a las velocidades lineales de las ruedas derecha e izquierda. Además el AmigoBot cuenta con ocho sonares frontales y dos traseros.

3.3 Software

Cada AmigoBot cuenta con un microprocesador Hitachi H8 a 20MHz como cerebro de su microcontrolador. Además cuenta con memoria FLASH de solo-lectura que tiene almacenado el sistema operativo. AmigOS se encarga de controlar todos los sistemas de bajo nivel y la electrónica del robot móvil, incluyendo las lecturas de los sonares y encoders, el ajuste de las velocidades de los motores, la reproducción de sonidos predefinidos y demás.

El AmigoBot es controlado mediante el lenguaje de programación C++ utilizando el entorno de desarrollo ARIA, en una PC con sistema operativo Windows 7.

3.4 Modelo Cinemático del uniciclo

Un AmigoBot es un robot de tipo uniciclo. Considerando movimiento sobre un plano, su posición y orientación quedan completamente definidas por las variables de estado (x, y, θ). Donde x, y representan la posición del robot sobre el plano y θ su ángulo de orientación con respecto al eje X. Las derivadas de primer orden respecto al tiempo de estas variables se relacionan con la velocidad lineal v y la velocidad angular w mediante el modelo cinemático. En la Fig. 6 se muestra el diagrama cinemático de un AmigoBot.

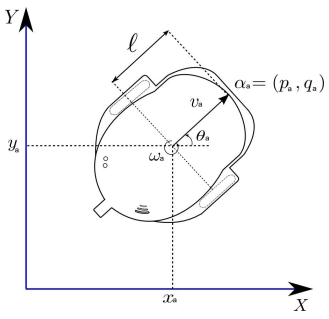

Figura 6. Diagrama cinemático de un AmigoBot

Del diagrama cinemático obtenemos el modelo matemático siguiente

$$\begin{bmatrix} \dot{x}_a \\ \dot{y}_a \\ \dot{\theta}_a \end{bmatrix} = \begin{bmatrix} \cos\theta_a & 0 \\ \sin\theta_a & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_a \\ w_a \end{bmatrix} = \bar{A}_a(\theta_a) \begin{bmatrix} v_a \\ w_a \end{bmatrix} \tag{19}$$

Es bien conocido que el sistema (19) no puede ser estabilizado mediante leyes de control continuas e invariantes en el tiempo. Debido a lo anterior se estudiara la cinemática del punto α_i.
El punto α_i se encuentra a una distancia ℓ del centro del eje de las llantas y sus coordenadas están dadas por:

$$\alpha_a = \begin{bmatrix} p_a \\ q_a \end{bmatrix} = \begin{bmatrix} x_a + \ell\cos\theta_a \\ y_a + \ell\sin\theta_a \end{bmatrix} \tag{20}$$

Calculando la cinemática del punto α_i se tiene:

$$\dot{\alpha}_a = \begin{bmatrix} \dot{x}_a - \ell\sin\theta_a \cdot \dot{\theta}_a \\ \dot{y}_a + \ell\cos\theta_a \cdot \dot{\theta}_a \end{bmatrix} \tag{21}$$

$$\dot{\alpha}_a = \begin{bmatrix} \cos\theta_a & -\ell\sin\theta_a \\ \sin\theta_a & \ell\cos\theta_a \end{bmatrix} \begin{bmatrix} v_a \\ w_a \end{bmatrix} = \hat{A}_a(\theta_a) \begin{bmatrix} v_a \\ w_a \end{bmatrix} \tag{22}$$

Debido a que, $\det(\hat{A}_a(\theta_a)) \neq 0 \ \forall\theta_a$, la matriz de desacoplamiento $\hat{A}_a(\theta_a)$ es no singular lo que nos permite establecer una estrategia de control para el punto α_a.

4. SISTEMA DE LOCALIZACIÓN EXTERNO

Para determinar la posición de ambos robots en el espacio se utilizó el software Motive junto con 12 cámaras de la marca OptiTrack, ver Fig 7 . Para la localización del drone se le colocaron 5 marcadores reflectantes como se muestra en la Fig. 8. Dado que Motive solo se encuentra disponible para sistemas operativos basados en Windows los datos son obtenidos en una computadora con sistema operativo Windows 7 y posteriormente enviados por medio de un ruteador de banda ancha SMC Barricade a la computadora que se encarga de controlar al drone, en el sistema GNU/Linux Ubuntu, utilizando el protocolo de comunicación TCP/IP. Para la simulación de modelos

Figura 7. Cámaras Optitrack

matemáticos y el análisis de resultados experimentales se utilizo el software Matlab de Matworks.

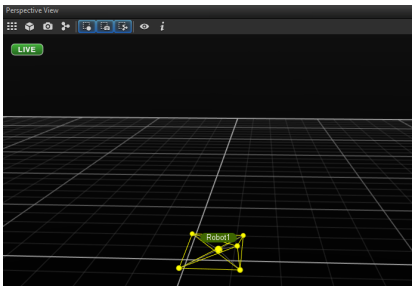

Figura 8. Disposición de los marcadores reflectantes en el drone

5. CONTROL MULTI-AGENTE PARA AGENTES HETEROGÉNEOS

En esta sección se diseña e implementa en tiempo real una ley de control para resolver el problema de marcha utilizando agentes heterogéneos. La ley de control está basada en un control de marcha con vector de formación variante en el tiempo. El tipo de esquema utilizado es el de líder-seguidor, en el cual el agente líder, el AmigoBot R_a, debe seguir una trayectoria dada y el agente seguidor, el drone R_d, debe mantener un vector de formación con respecto al agente líder.El agente líder desconoce la posición de

los demás robots. El robot seguidor conoce su propia posición y la posición del líder. La trayectoria que el líder deberá seguir esta dada por una Lemniscata de Gerono (caso especial de las curvas de Lissajous), cuyas ecuaciones paramétricas son:

$$m_x(t) = c_x + a_x \cos(2\pi t/T) \quad (23)$$
$$m_y(t) = c_y + a_y \sin(2\pi t/T)\cos(2\pi t/T) \quad (24)$$
$$\dot{m}_x(t) = -2a_x\pi \sin(2\pi t/T)/T \quad (25)$$
$$\dot{m}_y(t) = 2a_y\pi \cos(4\pi t/T)/T \quad (26)$$

Se eligió dicha trayectoria debido a que es infinitamente diferenciable. Se ajustaron los parámetros de la curva para una Lemniscata de 2,4 m de largo (eje X) y 3,0m de ancho (eje Y) centrada en el origen. El periodo de la trayectoria es $T = 60$s.

5.1 Ley de control para el líder

El líder de la formación es el AmigoBot. En esta sección se desarrolla la ley de control, basada en el modelo cinemático definido anteriormente, que permite el control de marcha para el AmigoBot. Se busca controlar el punto α del robot. A partir de la ecuación (22), definimos la ley de control:

$$\begin{bmatrix} v_a \\ w_a \end{bmatrix} = A_a^{-1}(\theta_n) \begin{bmatrix} u_{a1} \\ u_{a2} \end{bmatrix} \quad (27)$$
$$\begin{bmatrix} u_{a1} \\ u_{a2} \end{bmatrix} = \begin{bmatrix} \dot{m}_x(t) - k_a(\alpha_{ax} - m_x(t)) \\ \dot{m}_y(t) - k_a(\alpha_{ay} - m_y(t)) \end{bmatrix} \quad (28)$$

donde $m_x(t)$ y $m_y(t)$ son las componentes de la trayectoria a seguir y $\dot{m}_x(t)$ y $\dot{m}_y(t)$ son las componentes de la velocidad de dicha trayectoria definida por las ecuaciones (23)-(26). Obtenemos la ley de control:

$$\begin{bmatrix} v_a \\ w_a \end{bmatrix} = A_a^{-1}(\theta_n)\left[\dot{m}(t) - k_a(\alpha_a - m(t))\right] \quad (29)$$

Dada la ley de control, se requiere comprobar que los errores de posición convergen a cero. Para lo que definimos el error de posición y su dinámica:

$$e_a = \alpha_a - m(t) \quad (30)$$
$$\dot{e_a} = \dot{\alpha}_a - \dot{m}(t) \quad (31)$$

De la ecuación (28) obtenemos que:

$$\dot{e}_a = -k_a(e_a) \quad (32)$$

Lo cual demuestra que el error converge a cero de manera exponencial.

5.2 Ley de control para el seguidor

Es interesante notar que el modelo del AR Drone 2.0 dado por la ecuación (18) puede ser controlado como un robot planar omni-direccional, lo que significa que puede moverse en cualquier dirección con un ángulo de guiñada constante o variable. Se controlará la posición en el plano XY del punto central del drone. A partir de la ecuación (18) definimos la ley de control

$$\begin{bmatrix} \theta \\ \phi \end{bmatrix} = A_d^{-1}(\psi) \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = \frac{1}{g} \begin{bmatrix} -\cos(\psi) & -\sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{bmatrix} \begin{bmatrix} u_{d1} \\ u_{d2} \end{bmatrix} \quad (33)$$

Definiendo:

$$\begin{bmatrix} u_{d1} \\ u_{d2} \end{bmatrix} = \begin{bmatrix} \ddot{\alpha}_x^*(t) - k_v(\dot{x} - \dot{\alpha}_x^*(t)) - k_p(x - \alpha_x^*(t)) \\ \ddot{\alpha}_y^*(t) - k_v(\dot{y} - \dot{\alpha}_y^*(t)) - k_p(y - \alpha_y^*(t)) \end{bmatrix} \quad (34)$$

Donde $\alpha_x^*(t)$ y $\alpha_y^*(t)$ son las componentes del punto $\alpha^*(t)$ a seguir, $\dot{\alpha}_x^*(t)$ y $\dot{\alpha}_y^*(t)$ son las componentes de velocidad de dicho punto y $\ddot{\alpha}_x^*$ y $\ddot{\alpha}_y^*$ son las componentes de la aceleración del mismo. El punto α^* representa un la posición deseada del drone, la cual se determina a partir del punto α del AmigoBot más un cierto vector formación variante en el tiempo, es decir:

$$\alpha^*(t) = \alpha_a + C_{ad}(t), \quad (35)$$
$$\dot{\alpha}^*(t) = \dot{\alpha}_a + \dot{C}_{ad}(t), \quad (36)$$
$$\ddot{\alpha}^*(t) = \ddot{\alpha}_a + \ddot{C}_{ad}(t), \quad (37)$$

donde α_a es el la posición del punto frontal del AmigoBot líder y C_{ad} es el vector de posición variante en el tiempo que determina la formación entre el AmigoBot y el drone y se define como:

$$C_{ad}(t) = \begin{bmatrix} \cos(\theta_a) & -\sin(\theta_a) \\ \sin(\theta_a) & \cos(\theta_a) \end{bmatrix} c_{ad}, \quad (38)$$
$$\dot{C}_{ad}(t) = \begin{bmatrix} -\sin(\theta_a) & -\cos(\theta_a) \\ \cos(\theta_a) & -\sin(\theta_a) \end{bmatrix} c_{ad}, \quad (39)$$
$$\ddot{C}_{ad}(t) = \begin{bmatrix} -\cos(\theta_a) & \sin(\theta_a) \\ -\sin(\theta_a) & -\cos(\theta_a) \end{bmatrix} c_{ad}, \quad (40)$$

donde θ_a es el ángulo de orientación del AmigoBot y c_{ad} es un vector de posición constante. Dada la ley de control, se requiere comprobar que los errores de posición convergen a cero. Para lo cual se define el error de posición y la dinámica de dicho error:

$$e_d = \xi - \alpha^*(t) \quad (41)$$
$$\dot{e}_d = \dot{\xi} - \dot{\alpha}^*(t) \quad (42)$$
$$\ddot{e}_d = \ddot{\xi} - \ddot{\alpha}^*(t) \quad (43)$$

De la ecuación (34) se observa que:

$$\ddot{e}_d = -k_x(e_d) - k_v(\dot{e}_d) \quad (44)$$

Con lo que se concluye que el error converge a cero de manera exponencial. El ángulo deseado al que se debe posicionar el drone se obtiene mediante la ecuación:

$$\psi_d = \arctan 2\left(\frac{\dot{\alpha}_y^*(t)}{\dot{\alpha}_x^*(t)}\right) \quad (45)$$

Para realizar el control del ángulo ψ se propone el control de tipo proporcional mostrado en la ecuación (46), debido a que la velocidad de respuesta del controlador interno del drone es muy rápida

$$u_3 = -k_\psi(\psi - \psi_d) \quad (46)$$

Debido a que el drone solamente conoce la posición real del AmigoBot y no su velocidad o aceleración es necesario estimar la velocidad y aceleración del AmigoBot, dicha estimación se realiza por medio de una derivada "sucia" aproximada a partir de 3 puntos.

6. EXPERIMENTOS EN TIEMPO REAL

Se realizo la prueba experimental de marcha utilizando las cámaras optitrak y los sensores respectivos a cada robot como retroalimentación. Cada robot cuenta con su propia computadora de control que obtiene los datos de las posiciones calculadas por las cámaras optitrack mediante un programa servidor-cliente que se conecta con todas las computadoras de control mediante un ruteador utilizando el protocolo TCP/IP. El primer experimento consiste en que el drone siga al AmigoBot con un vector

de posición $c_{ad} = [-0{,}5, 0]^T$, mientras que el AmigoBot sigue la trayectoria dada por las ecuaciones (23)-(26). La Fig. 9 muestra la trayectoria seguida por el líder y por el seguidor. En dicha figura se observa que el agente seguidor sale de la trayectoria seguida por el líder en algunas zonas de la trayectoria, esto se debe al fenómeno sobreviraje provocado por el agente seguidor al buscar mantener en todo momento el vector de posición C_{ad} con respecto al líder. En las Figs. 10-13 se observan las componentes en

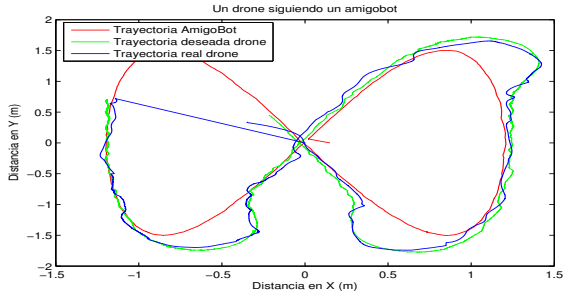

Figura 9. Resultado experimental un amigobot como líder y un drone como seguidor

los ejes X y Y del error entre la trayectoria deseada y la trayectoria real del robot seguidor.

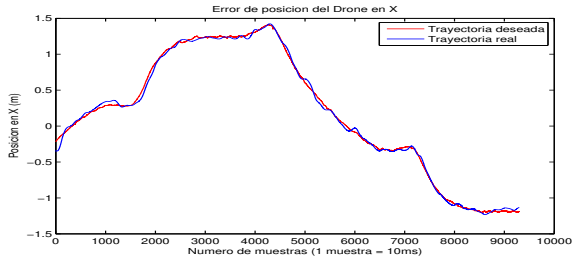

Figura 10. Trayectoria a seguir en X por el drone

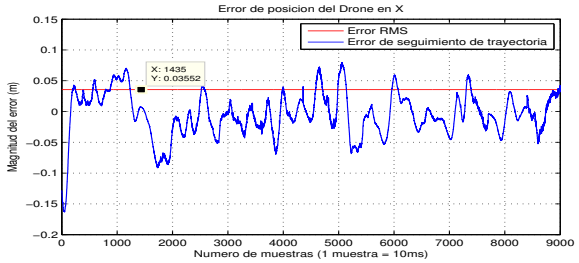

Figura 11. Error de seguimiento del drone en X

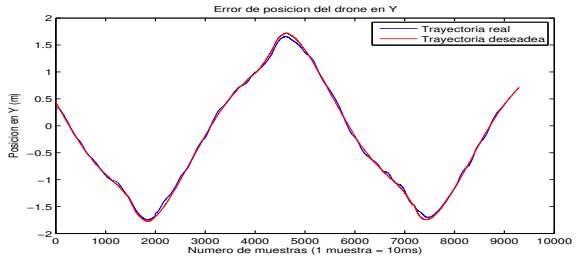

Figura 12. Trayectoria a seguir en Y por el drone

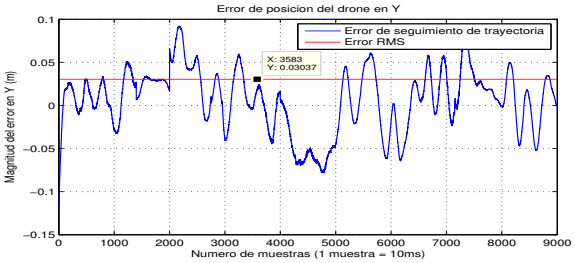

Figura 13. Error de seguimiento del drone en Y

7. CONCLUSIONES Y TRABAJO FUTURO

En este artículo se consideró el problema de seguimiento de trayectorias para un sistema multi-agente heterogéneo conformado por 2 robots móviles diferentes. Se obtuvieron leyes de control para cada robot basadas en los modelo matemáticos de los mismos. Las pruebas experimentales mostraron la efectividad de las leyes de control propuestas, al ser aplicadas a los robots. Como trabajo futuro se propone realizar experimentos con un mayor número de agentes, además de reducir el sobreviraje.

REFERENCIAS

Arai, T., Pagello, E., Parker, L. E. Guest editorial advances in multirobot systems, IEEE Transactions on Robotics and Automation Vol. 18(No. 5): 655–661.(2002)

Asahiro, Y., Asama, H., Suzuki, I., Yamashita, M. Improvement of distributed control algorithms for robots carrying an object, International Conference on Systems, Man, and Cybernetics, IEEE, Tokyo, Japan, pp. 608–613.(1999)

Cao, Y. U., Fukunaga, A. and Kahng, A Cooperative mobile robotics: Antecedents and directions Autonomous Robots Vol. 4(No. 1): 7–27.(1997).

Chen, Y. Q. and Wang, Z. Formation control: A review and a new consideration, International Conference on Intelligent Robots and Systems, IEEE/RSJ, Edmonton, Canada, pp. 3181–3186.(2005)

Chopra, N., Spong, W. Output synchronization of nonlinear systems with relative degree one. Adv. Learning Control, LNCIS371:51–64. (2008).

Do, K. Formation tracking control of unicycle-type mobile robots, International Conference on Robotics and Automation, IEEE, Roma, Italy, pp. 2391–2396.(2007)

García Carrillo, L.R., Dzul López, A.E., Lozano, R., Pégard, C. Quad Rotorcraft Control, Springer, 179 p, ISBN 978-1-4471-4398-7 (2013)

Jönsson, U.T., Kao, C.-Y. Consensus of heterogeneous linear agents applied to a formation control problem. In 49th IEEE Conference on Decision and Control. (2010)

Olfati-Saber, R. and Murray, R. Distributed cooperative control of multiple vehicle formations using structural potential functions The 15th IFAC World Congress, IFAC, Barcelona, Spain, pp. 2864–2869.(2002)

Yamaguchi, H. A distributed motion coordination strategy for multiple nonholonomic mobile robots in cooperative hunting operations Robotics and Autonomous Systems Vol. 43(No. 1): 257–282.(2003)

Zheng, Y., Zhu, Y., Wang, L. Consensus of heterogeneous multi-agent systems. IET Control Theory and Applications, 5:1881–1888. (2011).