Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática Industrial 21 (2024) 125-136

www.revista-riai.org

Muestreo y comunicación: impacto en el control de formaciones en sistemas multi-robot heterogéneos

Francisco-José Mañas-Álvarez*, María Guinaldo, Raquel Dormido, Sebastián Dormido

Departamento de Informática y Automática, E.T.S.I. Informática, Universidad Nacional de Educación a Distancia (UNED), Madrid, España.

To cite this article: Mañas-Álvarez, F.J., Guinaldo, M., Dormido, R., Dormido, S. 2024. Sampling and and communication: impact on formation control in heterogeneous multi-robot systems. Revista Iberoamericana de Automática e Informática Industrial 21, 125-136. https://doi.org/10.4995/riai.2023.20155

Resumen

Este trabajo presenta el análisis del efecto de la frecuencia de muestreo y comunicación en un sistema multi-robot (SMR) en su desempeño temporal y en la carga computacional. El sistema experimental está compuesto por robots móviles del tipo Khepera IV y robots aéreos del tipo Crazyflie 2.1. El análisis se realiza sobre el movimiento del SMR desde unas condiciones iniciales hasta una formación deseada, que se define en base a un conjunto de distancias relativas entre agentes. Se evalúan tres escenarios en relación a la arquitectura del nivel de control: centralizado, distribuido en ROS 2 y distribuido a bordo del robot. Se determina la frecuencia mínima operativa para un muestreo periódico, y se presenta un protocolo de muestreo basado en eventos como propuesta para la reducción de transmisiones de mensajes. Para este caso, se determina el umbral constante óptimo que, con un desempeño temporal equivalente al mejor muestreo periódico, obtiene una reducción del muestreo de un 80 %.

Palabras clave: Sistema Multi-Robot, Control de Formación, Control basado en eventos.

Sampling and and communication: impact on formation control in heterogeneous multi-robot systems

Abstract

This work presents the analysis of the sampling and communication frequencies in a multi-robot system (MRS) and its effect over the temporal performance and computational load. The experimental system is composed of mobile robots (the Khepera IV), and a type of aerial robots, the Crazyflie 2.1. The analysis is performed for the formation control of the MRS from initial conditions to a desired formation, which is defined in terms of relative distances between agents. Three scenarios regarding the control architecture are evaluated: centralized, distributed in ROS 2, and distributed onboard the robot. The minimum operating frequency for periodic sampling is determined, and an event-based sampling protocol is presented to reduce the number of transmitted messages. In this case, the optimal constant threshold is determined that, with a temporal performance equivalent to the best periodic sampling, obtains a reduction in the number of samples of around 80 %.

Keywords: Multi-Robot System, Formation control, Event-based control.

1. Introducción

En el paradigma actual de la sociedad "conectada" la integración entre los sistemas digitales y el mundo físico ha supuesto uno de los principales frentes en el desarrollo tecnológico que ha derivado en una eclosión de los sistemas ciberfísicos (SCF). Un SCF integra de forma transparente y eficaz procesos físicos, computación y comunicación, permitiendo a través de redes de sensores y actuadores monitorizar y/o controlar componentes o procesos físicos en distintos dominios de aplicación (Bogdan and Marculescu, 2011).

Uno de los principales ámbitos de desarrollo en auge de los SCF es la robótica móvil, particularmente la de tipo colaborativo. La demanda creciente de soluciones autónomas, flexi-

*Autor para correspondencia: fjmanas@dia.uned.es

bles y conectadas requiere la implementación de sistemas cada vez más complejos. En muchos casos, los requisitos que debe cumplir la solución aportada pueden ser cubiertos por un único agente. No obstante, la complejidad que puede requerir el diseño de este agente o el coste computacional asociado a la ejecución de la tarea requerida puede hacer la solución inviable. En este sentido, los Sistemas Multi-Robot, SMR (Cortés and Egerstedt, 2017) permiten mejorar la eficiencia en el procesamiento de datos gracias a que múltiples robots pueden tener objetivos compartidos y ejecutar colectivamente tareas más complicadas que las que podría completar un único robot.

Uno de los problemas que se debe abordar en aplicaciones de tipo cooperativo en un SMR es el movimiento coordinado entre robots. Hay múltiples trabajos relacionados con este tema, por ejemplo, sobre muestreo, monitorización, o vigilancia (Leonard et al., 2007; Fidan et al., 2007; Aranda et al., 2015). En todas estas tareas, mantener una formación dada juega un papel fundamental. En la literatura se pueden encontrar diferentes maneras de abordar el problema. Estas se pueden clasificar en función de la arquitectura de control, de las capacidades de medida y actuación que tengan los robots, del esquema de control (líder-seguidor, basadas en comportamiento, técnicas de estructura virtual), o de si la formación deseada se define de manera explícita o no (véase Oh et al. (2015) y sus referencias).

Desde el punto de vista de la arquitectura de control, se pueden encontrar esquemas centralizados, descentralizados o distribuidos (Guinaldo et al., 2017). En el caso centralizado, existe una entidad coordinadora que se encarga de recibir la información de todos los robots, la procesa, toma las decisiones y distribuye los comandos a los agentes. En el caso descentralizado, son los propios agentes los que toman sus decisiones, pero sin necesidad de una coordinación con el resto. Este aspecto está muy relacionado con las capacidades de medida de los agentes y con cómo se define la formación ya que, si poseen un sistema de posicionamiento absoluto y la formación se define de forma global para cada agente, no es necesaria una coordinación. Finalmente, en una arquitectura distribuida, esa interacción con otros agentes es necesaria para alcanzar el objetivo de control, y en tal caso la formación deseada se define en términos de posiciones relativas (Ren and Sorensen, 2008) o basada en distancias (Krick et al., 2009). Mientras que un esquema centralizado puede resultar más eficiente, por ejemplo, en términos del volumen de datos transmitidos, una arquitectura distribuida resulta más robusta frente a fallos que puedan ocurrir en el sistema.

En este contexto, la frecuencia de cómputo o de comunicación tiene un valor muy relevante en términos de eficiencia en arquitecturas distribuidas. En el caso de sistemas alimentados por baterías, esto tiene un impacto en la duración de las mismas y, como consecuencia, en la autonomía del sistema. Por tanto, la gestión de recursos se convierte en un problema crítico. Por un lado, la frecuencia de muestreo está estrechamente relacionada con el ancho de banda en los SCF. Cuanto mayor es la frecuencia, mayor es el uso del canal de comunicación y debe estudiarse para evitar que el ancho de banda se convierta en cuello de botella para el sistema. Por otro lado, si el sistema tiene capacidad de procesamiento limitada, es necesario reducir la frecuencia de muestreo relajando las exigencias temporales de ejecución. Esta reducción se acentúa cuanto más complejo es el algoritmo de control. Es por lo tanto fundamental el análisis de la frecuencia de muestreo del sistema bajo estudio. En este sentido, las estrategias de control basadas en eventos suponen una herramienta de gran utilidad frente a otras soluciones activadas por tiempo (Heemels et al., 2012). Las estrategias de muestreo o control basadas en eventos cambian el paradigma clásico de operar de forma periódica por la intervención sólo cuando es necesario, es decir, cuando hay un cambio relevante en el estado del sistema. De esta forma, se ha demostrado que se reduce la trasmisión innecesaria de mensajes y el consumo energético asociado. Existen en la literatura diversas estrategias bajo el nombre de control basado en eventos, pero la idea básica consiste en que cuando el error del sistema alcanza cierto umbral, se activa el muestreo y se cierra el lazo de control (Aranda-Escolastico et al., 2020).

El principal objetivo de este trabajo es analizar el efecto de la estrategia de muestreo y comunicación en la realimentación en el problema de control de formación en un SMR, en el cual la formación deseada se define en términos de distancias relativas entre agentes. Para ello, se ha hecho uso de la plataforma experimental presentada en Mañas-Álvarez et al. (2023), que permite la experimentación con robots autónomos de distinta naturaleza y que ha sido implementada a través de ROS 2. En primer lugar, se considera un paradigma basado en tiempo, en el que progresivamente se va disminuyendo la frecuencia hasta un valor en el cuál se observa una pérdida de rendimiento del sistema significativa. Posteriormente, se presenta un protocolo basado en eventos con distintos valores para el umbral del error. Con la finalidad de ahondar más en el control de formación para determinar los umbrales de trabajo óptimos para cada protocolo, se analizan tres escenarios con tres arquitecturas del controlador distintas: centralizado, distribuido en ROS 2 y distribuido a bordo del robot. Los resultados experimentales obtenidos proporcionan tanto valores óptimos de frecuencia y de umbral para el caso periódico y basado en eventos respectivamente, como una demostración de que, para un desempeño similar del sistema, el control basado en eventos permite una reducción en la transmisión de mensajes en torno a un 80 %. También se analiza el impacto sobre la carga computacional.

Este trabajo se estructura de la siguiente manera. En la sección 2 se describen las principales características de la plataforma experimental empleada. En la sección 3 se describe el SMR que se va a utilizar para el análisis del problema de control de formación. En la sección 4 se detallan las características de las experiencias llevadas a cabo y el cálculo del índice de desempeño empleado. En la sección 5 se detallarán los resultados obtenidos. Finalmente, en la sección 6 se presentan las conclusiones del trabajo, así como algunas posibles líneas de actuación futuras.

2. Materiales y métodos

El desarrollo experimental de este trabajo se ha realizado en *Robotic Park* (Mañas-Álvarez et al., 2023), cuya vista se muestra en la Figura 1. Se trata de una plataforma de interior implementada en ROS 2, diseñada y enfocada al estudio de los SMR. El entorno de trabajo comprende un volumen de 3,6 $m \times 3{,}6\ m \times 2\ m$ (largo $\times$ ancho $\times$ alto). Este espacio se considera suficiente para experiencias de control y navegación

en interiores con un número medio de robots (hasta 20 combinando aéreos y móviles).

2.1. Características de Robotic Park

La heterogeneidad de los componentes que integran la plataforma es una de sus principales características. Entre los robots aéreos disponibles de tipo cuadricóptero están los *Crazyflie 2.1* (Giernacki et al., 2017) y los *DJI Tello* (Giernacki et al., 2022). Son robots similares desde el punto de vista de su construcción siendo el último una plataforma comercial con mayor duración de la batería. Entre los robots móviles destacan los *Khepera IV* (Soares et al., 2016) y los *Turtlebot 3 Burger* (Amsters and Slaets, 2020).

Figura 1: Plataforma experimental *Robotic Park*.

Robotic Park dispone además de cuatro sistemas de posicionamiento para la localización de agentes en el volumen de trabajo. Cada uno está basado en una tecnología diferente para cubrir un mayor espectro de tipologías de robots. Dichos sistemas se pueden clasificar en dos tipos, externos o internos, en función de si la estimación de la posición se realiza en el agente o en un procesador externo. Los sistemas de posicionamiento externos son un sistema basado en captura de movimiento *Vicon Tracker*, con una precisión de ±0,017 *mm* (Vicon Motion Systems, 2022) y un sistema basado en ultrasonidos *Marvelmind*, con una precisión de ±2,0 *cm* (Vágner et al., 2022). Los sistemas de posicionamiento internos disponibles son la odometría de precisión para el caso de los Khepera IV, y los sistemas propios de los Crazyflie, *LightHouse* y *Loco Positioning System*. El sistema *LightHouse* (Taffanel et al., 2021), con una precisión de ±1,0 *mm*, está basado en la medición del desfase entre ángulos de balizas de infrarrojos externas en cuatro fotodiodos colocados en el robot. El sistema *Loco Positioning System* (Grasso et al., 2022), con una precisión de ±1,7 *cm* está basado en la tecnología Ultra Wideband (UWB). El único sistema 100 % compatible con todos los robots es *Vicon Tracker*, no obstante, la colocación de marcadores de forma asimétrica en el espacio limitado de los Crazyflie, necesaria para su correcto funcionamiento, supone un desafío cuando el número de robots es elevado. Por lo tanto, el disponer de distintos sistemas de posicionamiento compatibles entre sí permite ampliar el rango de robots y experiencias que se pueden realizar.

La comunicación entre todos los componentes de la plataforma *Robotic Park* se realiza a través de ROS 2 (*Robot Operating System* 2). ROS 2 es la generación más reciente de ROS, el estándar de código abierto más empleado en robótica. La implementación modular de *Robotic Park* mediante ROS 2 facilita el desarrollo de experiencias con diferentes tipologías de comunicación o agentes. Su uso permite que todos aquellos componentes que se encuentren en la misma red local y dispongan de un nodo de ROS 2 puedan comunicarse entre sí fácilmente a través de tópicos o servicios sin necesidad de configuraciones adicionales. Esta es una de sus principales ventajas pues permite la implementación de arquitecturas de comunicación totalmente distribuidas en las que cada nodo es completamente independiente y no está ligado a un maestro global. Además posibilita que con el mismo sistema hardware, la conmutación de arquitecturas de control centralizadas en un solo nodo a distribuidas en cada robot sea configurable a nivel software, ya sea en el inicio de la experiencia o mediante la conmutación de controladores de forma dinámica a lo largo de la operación del sistema.

Además, ROS 2 permite implementar SMR de forma recursiva mediante el uso de *namespaces* (véase la Figura 2). Estos subespacios dentro de la red de ROS 2 permiten la ejecución de forma simultánea de nodos y variables idénticos asociados a robots de la misma tipología sin necesidad de cambiar manualmente sus nombres identificativos para evitar solapes entre componentes. De esta manera se facilita realizar experiencias escalando el número de agentes sin cambios significativos en los ficheros de configuración. De forma complementaria, para responder a las tendencias tecnológicas en auge actualmente, *Robotic Park* presenta un diseño que facilita la integración y cooperación entre elementos virtuales y físicos. De esta manera es posible llevar a cabo experiencias de realidad mixta y gemelos digitales con la colaboración entre agentes reales y virtuales (Mañas-Álvarez et al., 2022, 2023). Mediante el uso de este tipo de experiencias se puede explorar sistemas con mayor cantidad de agentes, potenciar la sensorización de los robots o analizar el rendimiento de herramientas de simulación frente a casos de estudio específicos con altos niveles de exigencia.

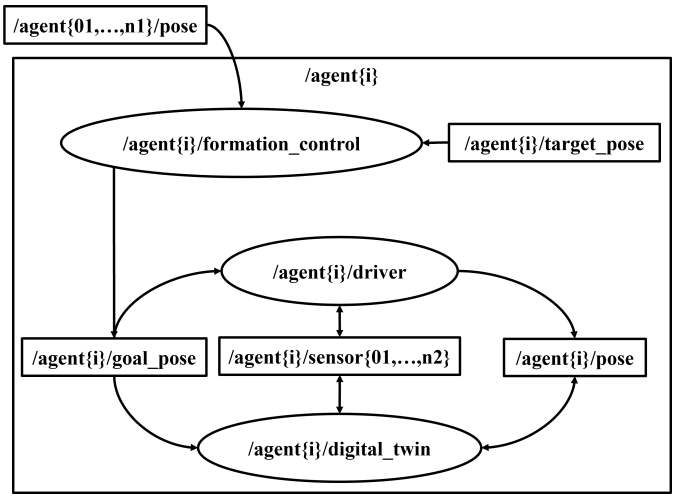


Figura 2: Ejemplo del *namespace* de un robot con gemelo digital y control de formación en un nodo de ROS 2.

2.2. Agentes

El robot **Crazyflie 2.1** (véase la Figura 3) es un robot aéreo del tipo cuadricóptero desarrollado por Bitcraze (Giernacki et al., 2017). Su tamaño y peso reducidos ($92mm \times 92mm \times 29mm$ y $27g$ respectivamente) enmarca al robot en la categoría de micro-cuadricópteros. Se trata de una plataforma de código abierto y bajo coste especialmente recomendable para experimentación en interiores. Esto la conforma como una herramienta con un gran alcance entre los usuarios profesionales y aficionados con un gran respaldo por la comunidad de desarrolladores existente. De hecho es fácil encontrar gran cantidad de proyectos que usan esta plataforma (Pichierri et al., 2023; Nguyen et al., 2022; Zekry et al., 2023).

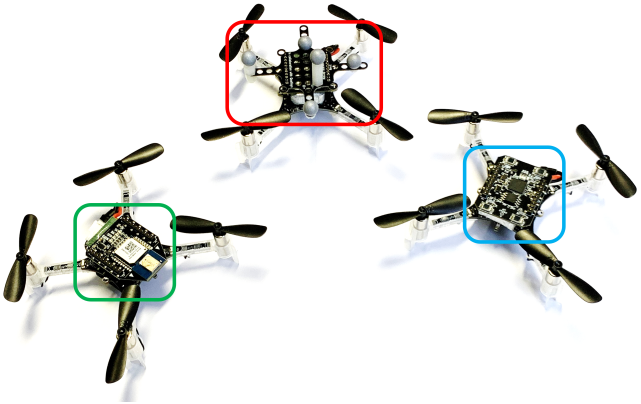


Figura 3: Crazyflies con las cubiertas para los sistemas *Loco Positioning* (izquierda, verde), *Vicon Tracker* (centro, rojo) y *Lighthouse* (derecha, azul).

Al tratarse de una plataforma de código abierto, se puede modificar el código del robot a todos los niveles, desde las aplicaciones de usuario de alto nivel hasta los controladores de bajo nivel que regulan el funcionamiento de los rotores. Dispone de librerías públicas gestionadas por la empresa desarrolladora a la que los usuarios pueden ir haciendo aportaciones. Entre estas librerías se encuentra disponible una API (*Application Programming Interface*) en Python para la conexión con los Crazyflies a través de la antena Crazyradio PA. El microcontrolador que implementa Crazyflie 2.1 es el STM32F405 que permite el incremento de la capacidad de cómputo y comunicación a través de la tarjeta de expansión *AI-deck* que cuenta con un procesador de bajo consumo GAP8.

Al tener un uso tan extendido, su modelo ha sido muy estudiado en la literatura (Silano et al., 2018; Garcia et al., 2017; Budaciu et al., 2019). De igual forma, es posible encontrar gran variedad de controladores aplicados a los diferentes niveles de control del robot como PIDs (Barra et al., 2020; Hasseni et al., 2021), predictivos (Didier et al., 2021), redes neuronales (Bansal et al., 2016) o combinaciones de diferentes arquitecturas (Chee et al., 2022). La arquitectura clásica de control a bajo nivel de los Crazyflies suele tener dos niveles formados cada uno por un sistema en cascada, tal y como se muestra en la Figura 4. El nivel superior tiene una frecuencia típica de 100 Hz y es el responsable de controlar la posición y la velocidad del robot. Su señal de entrada es la posición objetivo (q_i) y sus salidas son el empuje (Ω), los ángulos de cabeceo y balanceo (θ, ϕ) y la velocidad de referencia del ángulo de guiñada (r). El siguiente nivel es el control de orientación de los ángulos de cabeceo y balanceo, tanto en posición como en velocidad junto con el de velocidad del ángulo de guiñada. Este nivel requiere una frecuencia de funcionamiento mayor, 500 Hz, y sus señales de salida son los comandos que deben recibir los rotores del robot ($\omega_{1,2,3,4}$).

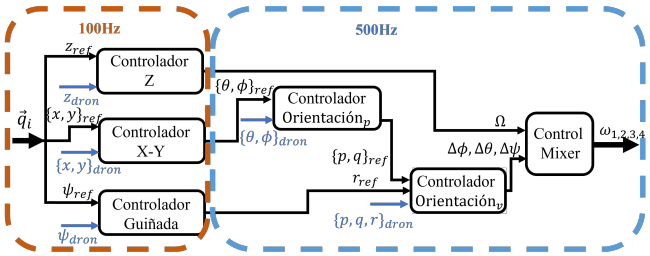


Figura 4: Diagrama de bloques de la arquitectura de control típica a bajo nivel de un Crazyflie 2.X.

El robot **Khepera IV** es la cuarta generación de robots móviles de acción diferencial desarrollados por K-Team (Soares et al., 2016). Por su tamaño reducido y características mecánicas su operación es especialmente recomendable en superficies duras y planas en interiores. Dispone de una sensorización con una alta precisión, que en el caso de la odometría, puede competir con gran parte de los sistemas de posicionamiento externos disponibles. Incorpora un procesador ARM Cortex A8 y un microcontrolador adicional para la gestión de periféricos. Opera un sistema Linux con distribución Yocto 1.8 y entre sus conexiones dispone de una conexión USB 2.0, una Wi-Fi 802.11 b/g y una Bluetooth 2.0 EDR.

Los robots diferenciales son plataformas muy usadas por su versatilidad y su modelo simple (Lamraoui et al., 2017). Se pueden encontrar en la literatura muchos de los controladores desarrollados para ellos como controladores PID tradicionales (Vázquez et al., 2021), LQR (Cornejo et al., 2018), lógica difusa (Štefek et al., 2021) o combinaciones de estos (Heikkinen et al., 2017). La arquitectura convencional de control de posición de estos robot está compuesta por dos niveles de control, tal y como se muestra en la Figura 5, precedidos por la conversión de la posición objetivo (q_i) al marco de referencia local (p_i) .

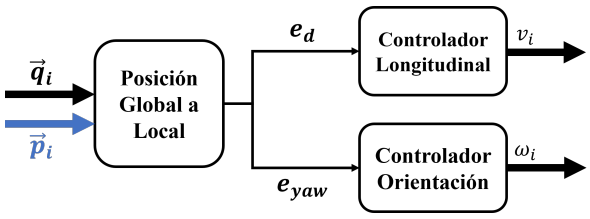


Figura 5: Diagrama de bloques de la arquitectura de control de bajo nivel implementada en el robot Khepera IV.

Por un lado, el control de desplazamiento longitudinal es el responsable del desplazamiento del robot, recibiendo como consigna la distancia hasta la posición deseada (e_d) y generando una referencia de velocidad lineal (v_i). Por otro lado, el controlador de orientación es el encargado de orientar el robot hacia la posición objetivo para el correcto funcionamiento del controlador longitudinal. Este controlador recibe de consigna el ángulo

que debe girar el robot para ubicar en el el eje x local del robot la posición objetivo $\left(e_{yaw}\right)$ y genera como señal de control la velocidad angular del robot (ω_i). Ambas velocidades se combinan para generar la tensión de alimentación de los actuadores.

3. Sistema Multi-Robot

La dinámica de un SMR se puede representar mediante grafos que permiten definir la topología de comunicación entre agentes. El grafo $\mathcal{G}(\mathcal{V}, \mathcal{E})$ queda determinado por un conjunto finito de N vértices $\mathcal{V} = \{v_1, ..., v_N\}$ que representan los nodos o robots y por un conjunto finito de aristas $\mathcal{E} \subseteq V \times V$, cada una de las cuales une pares ordenados de vértices, que representan los vínculos entre ellos. Un buen diseño de estos sistemas requiere la correcta sinergia entre los diferentes niveles de control de cada agente, ya que la definición de vínculos inalcanzables puede provocar un comportamiento errático en el sistema. Este sería el caso, por ejemplo, de incompatibilidades entre las posiciones de los agentes o requerimientos temporales demasiado exigentes para sus controladores internos.

3.1. Arquitectura de control

Un buen desempeño en un SMR depende directamente del diseño, cálculo e implementación de los diferentes controladores implicados en la operación de todos los agentes. La arquitectura de control implementada en *Robotic Park* mostrada en la Figura 6, es de tipo jerárquico, lo que permite el cambio de controladores de una forma sencilla para la evaluación de su desempeño. La arquitectura de control está segmentada en dos niveles. El nivel inferior implementa el control de posición y orientación de cada uno de los robots. En la sección 2.2 se describen soluciones de este nivel. Debido a las exigencias temporales que deben mantener estos controladores para evitar retardos dominantes introducidos por los canales de comunicación, y con la intención de mantener la mayor autonomía e independencia en el movimiento individual, se implementa este nivel de forma distribuida en los microcontroladores de cada robot. Además, en este mismo nivel se implementa la estrategia de evasión de obstáculos como una desviación de la posición objetivo que recibe de referencia (Mañas-Álvarez et al., 2023).

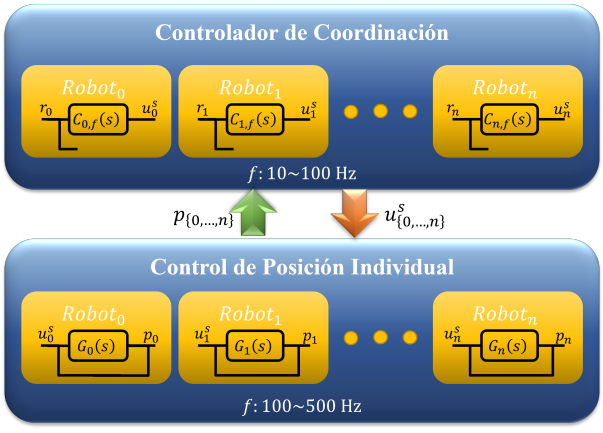


Figura 6: Diagrama de control jerárquico en Sistemas Multi-Robot.

El nivel de control superior es el responsable de coordinar el movimiento entre los robots que componen el SMR. Es, por tanto, el encargado de garantizar el cumplimiento de un objetivo de alto nivel definido para el sistema (por ejemplo, la adquisición de una formación), a partir de condiciones definidas a nivel local (por ejemplo, posiciones relativas o distancias deseadas entre los pares de agentes conectados a través del grafo).

En este trabajo, el grafo $\mathcal{G}$ se define de forma que se garantice la rigidez de una formación entre agentes (Anderson et al., 2008) y, como resultado, se generan distancias objetivo (d^*_{ij}) entre dos nodos cualesquiera v_i y v_j que estén conectados en el grafo $(v_i v_j \in \mathcal{E})$. Si $p_i \in \mathbb{R}^3$ es la posición del agente i, el control de formación implementado en este nivel se define según la siguiente ecuación:

$$u^s_i = -\sum_{j \in N_i} \mu_{ij}((d^*_{ij})^2 - \|p_i - p_j\|^2)(p_i - p_j), \qquad (1)$$

donde $\mu_{ij} > 0$ es la ganancia del controlador asociado al enlace entre los agentes i y j y $\|p_i - p_j\| = d_{ij}$ representa la distancia entre ellos. Normalmente, se puede considerar un valor igual para todas las ganancias, es decir, $\mu_{ij} = \mu,\ \forall i, j$. Si se desea que un agente tenga un mayor "peso" o dominancia sobre sus vecinos, la ganancia del controlador en estos últimos deberá ser mayor a las que implemente el primero. Obsérvese que cuando todas las distancias entre i y sus vecinos $j \in N_i$ convergen a los valores objetivo d^*_{ij}, la señal de control u^s_i se aproxima a cero.

El SMR en este nivel, independientemente de la ley de control que se utilice, permite diferentes arquitecturas, como pueden ser centralizadas, distribuidas dentro de la red de ROS 2 o distribuidas dentro de los microcontroladores de cada robot. Cada una de estas implementaciones presenta una serie de beneficios e inconvenientes que las hacen más adecuadas para aplicaciones específicas (Guinaldo et al., 2017). En el caso de las arquitecturas centralizadas, entre sus ventajas destacan la gestión síncrona de la acciones de control de todos los nodos del grafo y el reducido número de mensajes transmitidos en la red. En este caso, cada nodo asociado a un robot publica un mensaje, su posición, y se suscribe a un mensaje, su nueva posición objetivo. Esto implica una cantidad de mensajes de $n = 2N$ en la red, donde N es el número de nodos. Entre sus inconvenientes principales están la falta de autonomía en la operación del SMR, ya que ante un fallo en la máquina donde se ejecute el controlador todo el SMR deja de funcionar, la presencia de retardos dominantes a causa de los canales de comunicación puede condicionar la viabilidad de la ley de control usada, o problemas de rendimiento computacional cuando el sistema escala su cantidad de agentes.

Las soluciones distribuidas solventan gran parte de los inconvenientes de los sistemas centralizados al ofrecer una mayor flexibilidad en la red permitiendo que cada nodo pueda implementar de forma diferente su estrategia de control. De igual forma, ante un fallo en un nodo de control, no se interrumpe el funcionamiento del resto. Por otro lado, escalar el sistema es mucho más eficiente y predecible ya que el consumo de cada nodo de control es el mismo independientemente de la cantidad de nodos del grafo y el crecimiento es proporcional. Respecto a los retardos introducidos por la comunicación, se mantienen respecto al centralizado en el caso de la suscripción de los valores de posición de los vecinos conectados, pero se reducen en relación con el propio agente asociado al nodo del grafo. Por otro lado, la cantidad de mensajes transmitidos en la red

se ve incrementada significativamente. Esto se debe a que cada nodo publica la posición de su agente asociado y en lugar de suscribirse a su posición objetivo, debe suscribirse a todas las posiciones de sus vecinos, y por tanto $n = \sum_{i=1}^{N}(1 + |N_i|)$, donde $|N_i|$ representa el número de vecinos del agente i.

Por otro lado, los controladores distribuidos en los SMR pueden dividirse en dos subcategorías, los que implementan el controlador en la máquina que conecta el robot con la red de comunicación del sistema y los que realizan la implementación a bordo del robot en su microcontrolador. En ambos casos la cantidad de mensajes transmitidos en la red es la misma, pero se debe prestar especial atención a la comunicación entre el nodo de la red y el hardware del robot. La principal ventaja de las implementaciones a bordo del robot son la eliminación del retardo en la comunicación entre el robot y el controlador. No obstante, se debe estudiar que la carga computacional vinculada al controlador la pueda soportar el robot. De igual forma, se debe garantizar que el incremento de uso de la comunicación entre el nodo de la red y el hardware del robot no provoque un cuello de botella en el SMR.

3.2. *Muestreo basado en eventos*

Una de las principales características que se deben analizar durante el diseño de los SMR es la disponibilidad de recursos. Este aspecto es fundamental cuando el SMR incluye robots con limitaciones significativas, como es el caso de la duración de las baterías de los Crazyflie. En el caso del control de formación, la información que debe transmitirse por la red es la posición entre robots. Una gestión eficiente del uso de estos mensajes libera el canal de comunicación, permite incrementar el número de agentes en funcionamiento y reduce el consumo energético asociado a la transmisión de mensajes. De esta forma se amplía la duración de la batería de los robots y, en consecuencia, la duración de las experiencias en las que estén involucrados.

Además, al realizarse la implementación en un sistema digital, las transmisiones se deben realizar en instantes discretos de tiempo. En el muestreo convencional, estos eventos siguen un patrón periódico determinado por un tiempo de muestreo. Como alternativa a esta implementación, se ha demostrado que el muestreo basado en eventos asociado a variaciones en el estado del sistema supone un incremento en la eficacia debido a la reducción del número de muestras (Heemels et al., 2012).

En este caso, se define una función de disparo que, cuando se activa, indica que el sistema ha de ser muestreado. Esta función de disparo depende, en general, de una función de error y de un umbral, el cual puede ser constante o variar en función de otros parámetros. Cuando la función de error alcanza el valor del umbral definido, se genera un evento (muestreo). Normalmente, la función de error $e(t)$ se define como la norma de la diferencia entre la última medida transmitida $x(t_k)$ y la medida actual del estado $x(t)$. Así, el tiempo de disparo de un sistema de control basado en eventos se define de forma recursiva según

$$t_{k+1} = inf\{t : t > t_k, f(e(t), x(t)) > 0\}, \qquad (2)$$

donde $e(t) = x(t_k) - x(t)$ es el error de la función y $f(e(t), x(t))$ representa la función de disparo.

Para el marco experimental presentado en este artículo, definiremos la función de error para determinar cambios en la posición de cada agente. Dado que el posicionamiento de los agentes se realiza de forma interna en cada robot fusionando datos de los diferentes sensores disponibles (sistemas de posicionamiento externos, odometría, IMU, etc), la evaluación de la función de disparo se hace de forma periódica en la salida del estimador interno a la frecuencia de funcionamiento de este.

4. Desarrollo experimental

En este trabajo se pretende determinar la frecuencia de muestreo periódico mínima que permite mantener un desempeño adecuado en un problema de control en formación de un SMR, el umbral constante del muestreo basado en eventos máximo que obtenga un desempeño similar, y realizar una comparativa del número de mensajes transmitidos en la red para ambos casos. Para ello se emplea un SMR heterogéneo compuesto por cinco robots aéreos del tipo Crazyflie 2.1 y cuatro robots móviles diferenciales del tipo Khepera IV. El grafo no dirigido $\mathcal{G}(\mathcal{V}, \mathcal{E})$ que define este sistema está compuesto por el conjunto de nodos

$$\mathcal{V} = \{O, KH01, \ldots, KH04, CF01, \ldots, CF05\}, \qquad (3)$$

donde O es el origen de coordenadas, el prefijo "KH" representa a los robots del tipo Khepera IV y el prefijo "CF" a los robots del tipo Crazyflie 2.1. La representación tridimensional de la formación deseada se muestra en la Figura 7.

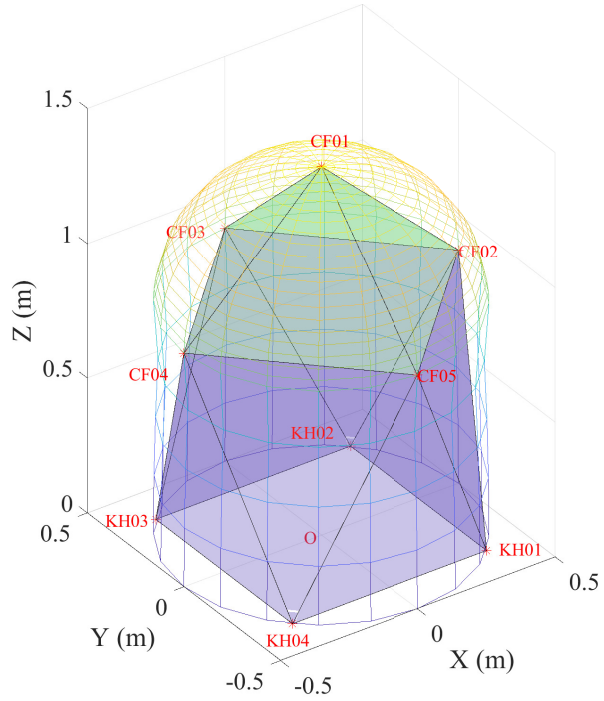


Figura 7: Representación gráfica tridimensional de la formación deseada.

La formación que se desea alcanzar se define del siguiente modo:

- Los robots aéreos deben mantener una distancia entre sí $d^*_{ij} = 0{,}7071\ m$ y esa misma distancia se debe mantener entre los robots móviles.
- En el caso de las distancias entre los robots móviles y los aéreos, este valor asciende hasta los $d^*_{ij} = 0{,}9468\ m$.

- Finalmente, respecto al origen de coordenadas, los robots móviles deben mantener una distancia $d^*_{i0} = 0{,}5\ m$, los robots $CF02, \cdots, CF04$ una distancia $d^*_{i0} = 1{,}0\ m$ y el robot $CF01$ una distancia $d^*_{i0} = 1{,}36\ m$.

Con el fin de realizar un estudio amplio del comportamiento en el control de coordinación de los SMR en los movimientos de formación, se analizan tres escenarios diferentes sobre cada una de las arquitecturas de comunicación permitidas en el segundo nivel de control descrito en la sección 3.1:

1. **Centralizado.** En este escenario todo el nivel de control de coordinación se ejecuta en un único nodo de ROS 2 independiente de los robots. Este nodo se suscribe a las posiciones de cada robot, ejecuta un controlador para cada uno y genera las consignas de referencia para sus correspondientes controles de posición individual publicándolas a la red en forma de tópicos para ser leídas por cada robot.

2. **Distribuido en ROS 2.** En este escenario cada robot posee su propio controlador de coordinación. Este controlador se ejecuta en un nodo diferente para cada robot. En nuestro caso la implementación se ha realizado en el mismo nodo con el que cada robot se comunica con el resto de la red de ROS2, por tratarse de la opción más eficiente.

3. **Distribuido a bordo.** En este escenario el controlador de coordinación de cada robot se ejecuta en su microcontrolador, por lo que el uso de ROS 2 se limita como canal de comunicación de la información entre robots.

Para cada uno de los escenarios establecidos, se ha realizado un barrido en las ventanas de operación de muestreo periódico y basado en eventos. En el muestreo periódico se ha comenzado con una frecuencia máxima de 50 Hz, valor límite que admite la API de Python para leer el posicionamiento interno de los robots aéreos. La frecuencia mínima se ha establecido en 0,5 Hz tras observarse una degradación significativa en los índices de desempeño del sistema para todos los escenarios. Los casos estudiados con este protocolo analizan las frecuencias 50 Hz, 20 Hz, 10 Hz, 5 Hz, 1 Hz y 0,5 Hz.

Para la evaluación del protocolo basado en eventos se ha tomado como punto de partida un umbral de disparo de $1cm$. Este umbral se ha incrementado hasta un valor máximo de $8cm$. Los casos estudiados con este protocolo emplean los umbrales 1 cm, 2 cm, 4 cm, 6 cm y 8 cm respectivamente. En total se han evaluado 11 casos entre ambos protocolos que se han repetido 7 veces cada uno para obtener una mejor estimación del desempeño del SMR y observar la repetitividad de las experiencias en función de la desviación estándar de los resultados.

Los índices de comportamiento analizados para la evaluación cuantitativa del desempeño del SMR en las distintas experiencias realizadas son los siguientes:

- *Tiempo de establecimiento, t_e.* Tiempo empleado por la formación hasta permanecer en un límite del 5 % de error en la formación deseada respecto del error inicial.

- *Tiempo medio de muestreo, T.* Este índice representa el flujo medio de información relativo a las transmisiones de posición de cada agente a lo largo del experimento.

- *Integral del Error Absoluto, IEA.* Representa el error global de la formación ponderando todos los errores de forma equivalente a lo largo del tiempo.

- *Integral del Error Absoluto ponderado en el Tiempo, IEAT.* Este índice pondera el error global de la formación por el tiempo transcurrido desde el inicio de la experiencia. En los sistemas que utilizan entradas en escalón, como el caso de los SMR al iniciar una formación, el error inicial es siempre elevado. Para realizar comparaciones entre los distintos casos de cada escenario, es más relevante que los errores que se mantienen en el tiempo tengan un mayor peso que los errores iniciales.

- *% Uso de la CPU por parte de los nodos donde se ejecuta el controlador de formación, η.* Para determinar la idoneidad de una arquitectura o protocolo en sistemas con restricciones de recursos computacionales, medir su consumo es un factor clave. Se registrará el % de uso total de la CPU que se emplea en el sistema (contabilizando todos los núcleos de la CPU), % de uso del núcleo donde se ejecutan los nodos responsables de la comunicación con los Crazyflie y los Khepera IV.

5. Resultados

Todos los ensayos realizados han seguido la misma secuencia de instrucciones, posiciones iniciales y duración para poder realizar la comparativa entre ellos. Inicialmente todos los robots aéreos reciben la orden de despegue y se fija su punto de equilibrio en la cota $z = 0{,}7m$. En este nuevo estado estacionario, comienza el control de formación y se registran los $18s$ siguientes, tiempo suficiente para que el sistema alcance el nuevo estado estacionario. En la Figura 8 se muestran las trayectorias seguidas por el sistema (trazo rojo) y la formación final (trazo negro discontinuo) durante una de las experiencias realizadas.

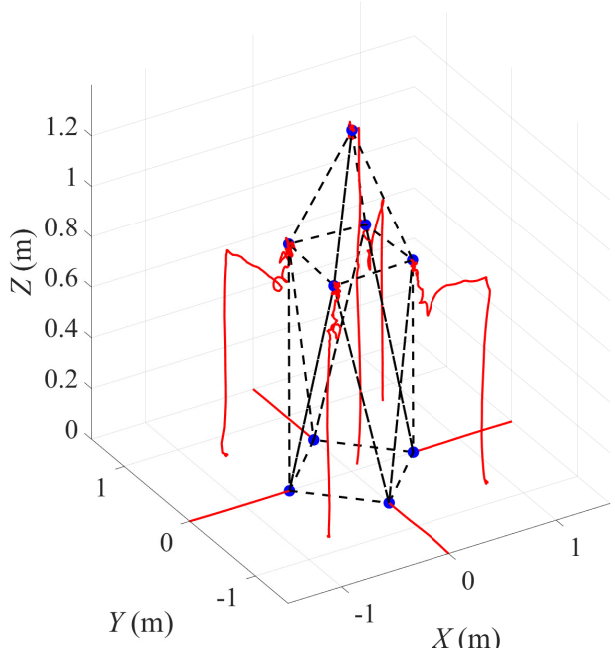
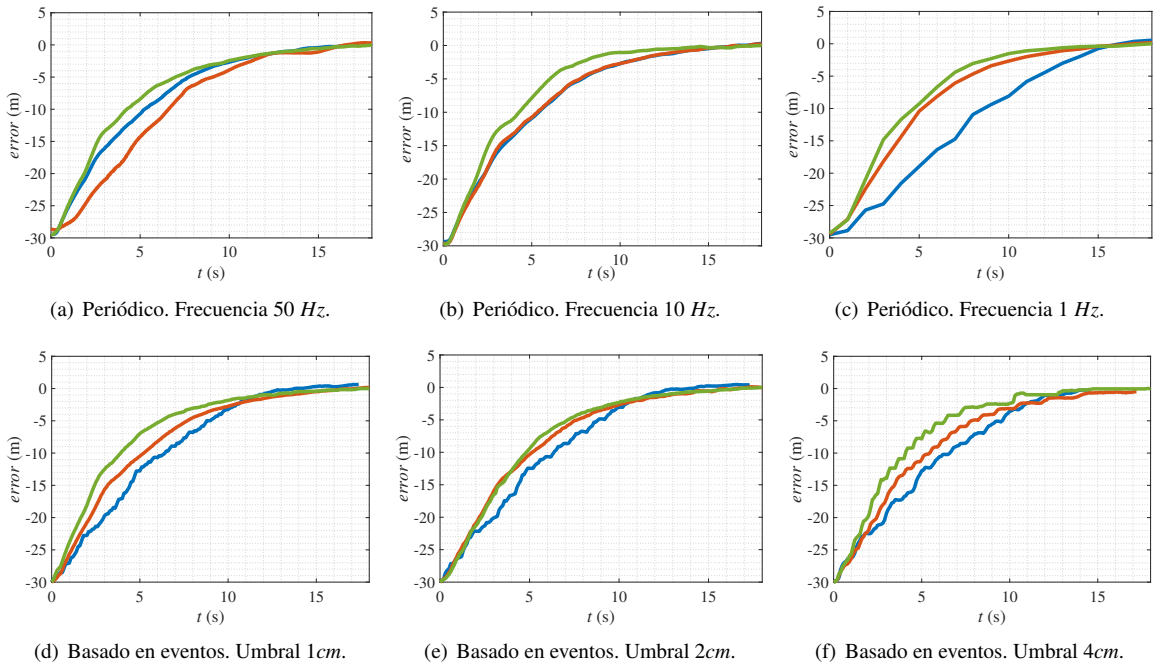


Figura 8: Trayectorias seguida por los robots con controlador centralizado y muestreo periódico a $10Hz$.



(a) Periódico. Frecuencia 50 Hz. (b) Periódico. Frecuencia 10 Hz. (c) Periódico. Frecuencia 1 Hz.

(d) Basado en eventos. Umbral 1cm. (e) Basado en eventos. Umbral 2cm. (f) Basado en eventos. Umbral 4cm.

Figura 9: Comparativa de respuestas temporales. Azul: Centralizado; Rojo: Distribuido en ROS 2; Verde: Distribuido a bordo.

En la Figura 9 se muestra la evolución temporal de los errores de formación en los casos más significativos para los muestreos periódicos y basados en eventos (evolución media entre todos los ensayos realizados). En un análisis cualitativo de los resultados se observa que en todos los casos el sistema converge al error nulo y la implementación distribuida con el controlador en los microcontroladores de los robots (trazo verde) ofrece una respuesta más rápida. El comportamiento de la respuesta se mantiene similar al cambiar las condiciones de muestreo. Para los escenarios de los controladores centralizados y distribuidos ejecutados en la red de ROS 2, las diferencias son más notables, especialmente en los casos de muestreo periódico.

En el caso del muestreo periódico a 50 Hz, Figura 9(a), el control centralizado presenta una respuesta más rápida que el distribuido. Esta diferencia desaparece en el muestreo a 10 Hz obteniendo respuestas prácticamente idénticas entre ambos escenarios (Figura 9(b)). Cuando la frecuencia continua decreciendo (Figura 9(c)), la degeneración obtenida en el desempeño se hace más significativa en el caso centralizado y los casos distribuidos alcanzan una evolución similar. En el caso del muestreo basado en eventos, la evolución observada es más homogénea entre los distintos umbrales para las diferentes implementaciones, siendo más acuciada la dinámica escalonada en la respuesta con el incremento del valor de los umbrales. Para estas implementaciones, la respuesta inicial más rápida la ofrece el sistema con los controladores a bordo del robot y la más lenta aquellos que emplean el controlador centralizado.

Mediante el análisis del tiempo de establecimiento, se determina de forma cuantitativa la velocidad en la respuesta del sistema. Los resultados promedios obtenidos para los dos tipos de muestreo y sus desviaciones estándar se reflejan en la Figura 10. Se observa cómo en el caso periódico, a partir de los 5 Hz el sistema empeora haciéndose más lento y con una mayor desviación en los resultados entre ensayos. Esto indica que la repetitividad entre experiencias empeora. Para el muestreo basado en eventos se observa una dinámica similar a los casos periódicos por encima de 5 Hz para los umbrales de 1 cm y 2 cm. Con umbrales mayores, la ralentización progresiva en la dinámica del sistema es equivalente en las tres implementaciones. Para completar el análisis de la respuesta temporal, se observan los índices IEA e $IEAT$, que indican la velocidad del sistema y su distribución en el tiempo.

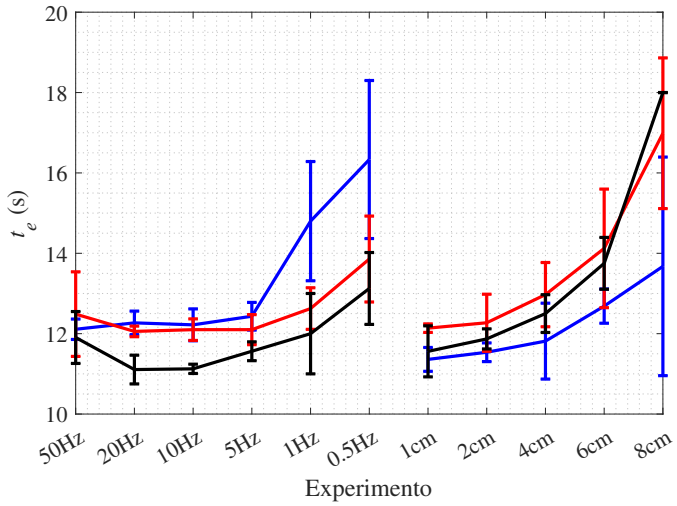
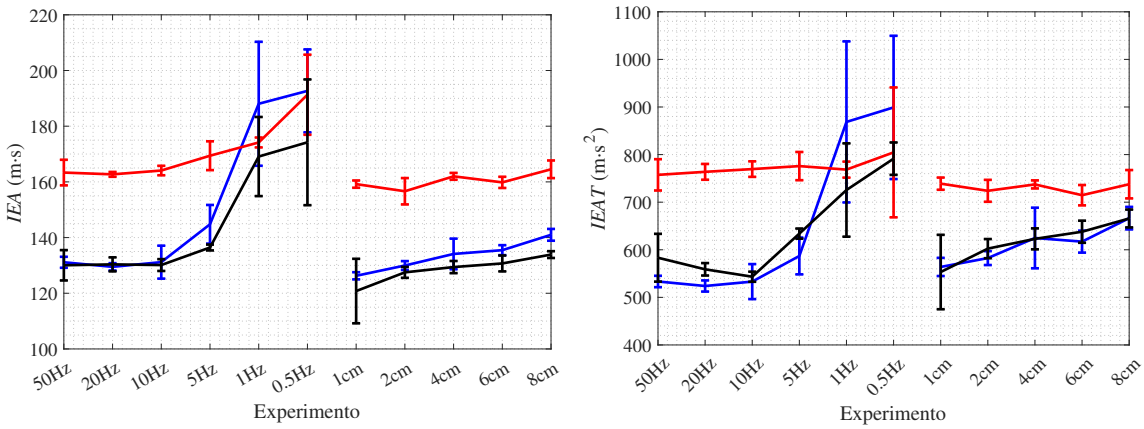


Figura 10: Tiempo de establecimiento de la formación. Azul: Centralizado; Rojo: Distribuido en ROS 2; Negro: Distribuido a bordo.

Los resultados estadísticos obtenidos se muestran en la Figura 11 para los dos tipos de muestreo. Para ambos índices, se observa una dinámica similar a la observada en el tiempo de establecimiento en lo que se refiere a los rangos de operación óptimos para ambas tipologías de muestreo. En la implementación de muestreo periódico, frecuencias de muestreo inferiores a los 5 Hz acusan un empeoramiento del desempeño significativo que justifican el rechazo de estas frecuencias para operar



(a) Integral del Error Absoluto.

(b) Integral del Error Absoluto ponderado en el Tiempo.

Figura 11: Resultados en índices de desempeño basados en la respuesta temporal. Azul: Centralizado; Rojo: Distribuido en ROS 2; Negro: Distribuido a bordo.

el sistema. En el caso del muestreo basado en eventos, las respuestas obtenidas son mucho más homogéneas con la variación de los parámetros y diferentes entre escenarios, no alcanzando un empeoramiento tan significativo como los obtenidos para las frecuencias periódicas de 1 Hz y 0,5 Hz. La homogeneidad observada es un indicador de una distribución equivalente de los errores del sistema a lo largo del tiempo. En términos globales, se observa que el mejor desempeño lo proporciona la implementación con el controlador a bordo y la peor con el controlador distribuido en los nodos de cada robot en ROS 2.

Tras analizar la respuesta temporal del SMR se analiza la diferencia en el consumo de recursos computacionales entre los diferentes casos estudiados. En la Figura 12 se muestra el % de uso de la CPU (contabilizándose todos los núcleos) donde se ejecuta el sistema. La tendencia observada sigue la evolución esperada acorde a las frecuencias de funcionamiento de los nodos en cada caso y el consumo asociado a la ejecución de controladores y publicación/suscripción de variables. En el caso centralizado y distribuido, se observan valores similares con una diferencia no mayor al 2 % en los casos más desfavorables. Para los casos basados en eventos, se observan unos valores homogéneos entre umbrales equivalentes al caso periódico a 5 Hz.

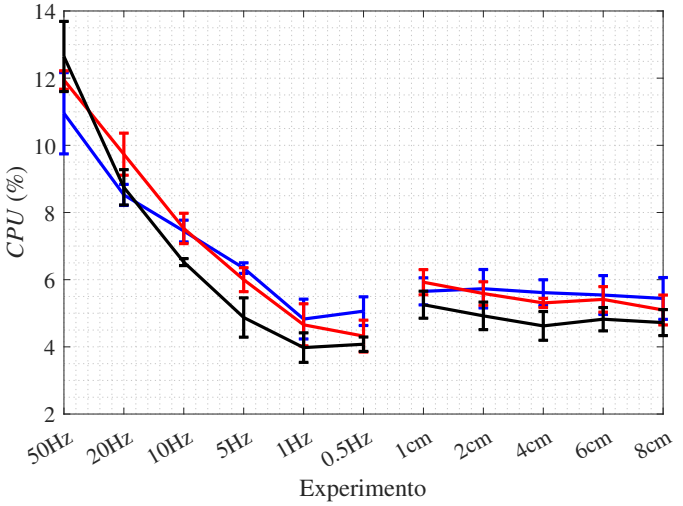
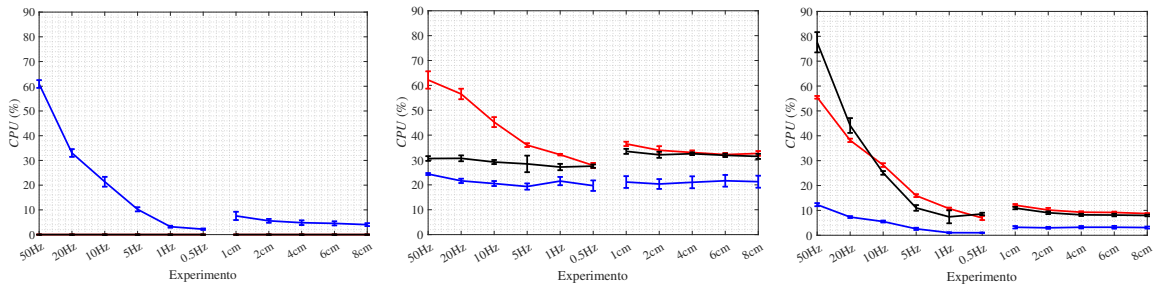


Figura 12: Porcentaje de uso de la CPU total.

Para profundizar más en los efectos de las comunicaciones y ejecución de los ciclos del controlador estudiado en la Figura 13 se aúnan los resultados obtenidos de la medición de los consumos aislados del nodo de control centralizado (Figura 13(a)), el nodo encargado de la comunicación con los Crazyflies (Figura 13(b)) y el conjunto de los cuatro nodos vinculados a los cuatro robots móviles del tipo Khepera IV (Figura 13(c)). Para el caso centralizado (trazo azul), se observa claramente la relación proporcional del consumo del controlador con la frecuencia de operación en el caso periódico y un comportamiento similar para el caso basado en eventos, siendo ligeramente superior en el menor umbral al generarse más eventos. En este escenario, el consumo de los nodos que se comunican directamente con los robots es muy homogéneo lo cual que significa que la lectura de posición del robot y envío de las consignas de posición tienen un efecto poco significativo en el consumo de recursos computacionales. En el caso del control distribuido pero ejecutado en los nodos de ROS 2 (trazo rojo), se observa cómo la dinámica observada en el controlador del caso centralizado se superpone antes con la registrada en los nodos de los robots. Nuevamente existe una relación proporcional en los casos de muestreo periódico entre el consumo de recursos y la frecuencia de muestreo. En el caso basado en eventos, se obtienen valores homogéneos con una ligera variación inversamente proporcional con el valor del umbral de disparo. En este escenario, la comunicación con los robots es la misma que en el caso del controlador centralizado. Finalmente, el último escenario implementa el controlador a bordo de cada robot (trazo negro). En este caso hay que señalar que se realiza un mayor uso del canal de comunicación con los robots ya que además de leer su posición se envía la posición de los cinco robots con los que se encuentra conectado cada agente. Esto queda reflejado en la diferencia medida con el escenario centralizado ya que en ambos casos el controlador no se ejecuta en el nodo. En el caso del nodo de comunicación con los Crazyflies, se observa un comportamiento semejante al escenario centralizado con un valor superior 10 puntos de media. En el caso de los Khepera IV, se observa que el consumo es superior a los escenarios anteriores en frecuencias periódicas por encima de los 10 Hz presentando una dinámica similar al caso distribuido en ROS 2 para frecuencias inferiores. Para los casos del muestreo basado



(a) Control de coordinación centralizado. (b) Crazyflies 2.1. (c) Kheperas IV.

Figura 13: Porcentaje de uso del núcleo de CPU por parte de los nodos que se comunican con los Crazyflies, con los Khepera IV y el controlador centralizado. Azul: Centralizado; Rojo: Distribuido en ROS 2; Negro: Distribuido a bordo.

en eventos, los datos reflejan, al igual que para los Crazyflies, que la dinámica de consumo sigue un comportamiento similar al caso distribuido con el controlador en ROS 2 con un valor medio superior en 7 puntos respecto del caso centralizado. Estos datos reflejan que la comunicación mediante el protocolo cliente-servidor que implementan los Kheperas tienen un consumo dominante a frecuencias superiores a los 10 Hz.

Finalmente, se examinan los diferentes escenarios y casos estudiados desde la perspectiva de la comunicación mediante el periodo de muestreo promedio (Figura 14). Este parámetro, en los casos de muestreo periódico, carece de interés por si mismo, puesto que se fuerza al sistema a operar a unos valores establecidos. No obstante, en las gráficas se muestran estos valores para establecer fácilmente la comparativa con los casos basados en eventos. Con la finalidad de extraer más información sobre el comportamiento del sistema, se analiza el valor del periodo de muestreo medio por separado para los conjuntos de los Kheperas y los Crazyflies. Se observa que el comportamiento de los Crazyflies (trazo discontinuo) es prácticamente idéntico para cualquiera de las implementaciones analizadas a lo largo de los distintos valores del umbral de disparo.

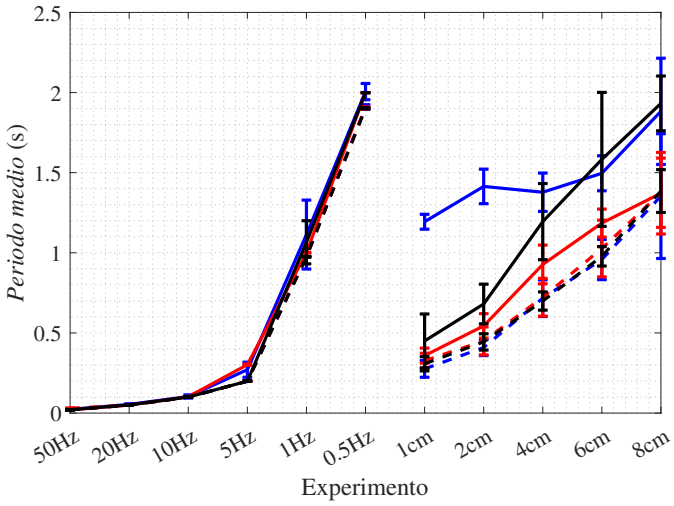


Figura 14: Periodo de muestreo medio. "–": Khepera IV; "––": Crazyflie 2.1; Azul: Centralizado; Rojo: Distribuido en ROS 2; Negro: Distribuido a bordo.

Por otro lado, los Kheperas (trazo continuo) sí presentan diferencias entre escenarios siendo las más significativas entre el control centralizado y los distribuidos en los dos umbrales de disparo menores. La progresión creciente sigue la tendencia esperada al aumentar el umbral de disparo. La implementación a bordo de los robots pierde relevancia a partir de un umbral de disparo de 4 cm dada su elevada desviación. En términos globales, se observa claramente que la reducción en la comunicación basada en eventos supone una mejora significativa frente al muestreo periódico por encima de los 5 Hz. Además, los robots móviles consiguen generación de eventos inferior a los robots aéreos bajo las mismas condiciones.

6. Conclusiones

En este trabajo se ha presentado un análisis del desempeño del control de coordinación encargado de alcanzar una formación objetivo en un Sistema Multi-Robot variando el protocolo de muestreo en la posición de sus agentes. Se han estudiado tres escenarios con diferentes implementaciones del controlador: centralizado, distribuido en ROS 2 y distribuido a bordo del robot. Dentro de cada escenario se han analizado un barrido de frecuencias para un protocolo de muestreo periódico y un barrido en el valor del umbral de disparo constante para un protocolo basado en eventos. Los resultados obtenidos se han evaluado desde tres perspectivas: la respuesta temporal del error de formación, el consumo de recursos computacionales y el uso del canal de comunicación. Estos enfoques son los principales aspectos clave que deben tenerse en cuenta en el diseño de SMR cuyos agentes presentan restricciones en la disponibilidad de recursos, energéticos o computacionales.

A la luz de los resultados obtenidos, se observa como la reducción de la frecuencia de muestreo perjudica el desempeño del sistema aunque reduce significativamente el consumo de recursos computacionales. Se determina que para alcanzar una solución de compromiso entre todos los resultados obtenidos, la mínima frecuencia de muestreo periódico que se puede aplicar a este tipo de sistemas sin perder rendimiento en el desempeño es de $10Hz$. De igual forma, se ha presentado una propuesta de muestreo basada en eventos para reducir el uso del canal de comunicación. Se ha observado cómo el mejor rendimiento para un sistema con umbral de disparo constate se obtiene para umbrales de valores 1 cm y 2 cm. Específicamente, para el caso basado en eventos con umbral de 2 cm frente a un muestreo periódico a 10 Hz se ha alcanzado una reducción en la comunicación del 89 % en el escenario centralizado, 80 % en el escenario distribuido en ROS 2 y 82 % en el escenario distribuido a bordo de los robots. En cuanto a los resultados obtenidos en

términos computacionales, se evidencia que los Crazyflies tienen una arquitectura hardware y protocolo de comunicación del robot con la red que ofrece un mejor rendimiento para implementar el controlador de coordinación en su propio microcontrolador. Por otro lado, los Kheperas ofrecen un mayor rendimiento cuando el control de coordinación se ejecuta en un nodo de la red de ROS 2. Por lo tanto, se demuestra que para Sistemas Multi-Robot heterogéneos como el estudiado en el presente trabajo, la implementación óptima no se puede alcanzar con una arquitectura única si no que se precisa de arquitecturas híbridas que aprovechen los puntos fuertes y suplan las debilidades de los agentes que componen el sistema. En este sentido, la plataforma *Robotic Park* está preparada para poder desarrollar este tipo de implementaciones y optimizar los resultados obtenidos en futuras experiencias.

Una vez establecidos experimentalmente los puntos óptimos de muestreo para el caso periódico y el caso más simple basado en eventos, se continuará trabajando en la implementación de mejoras que permitan incrementar la optimización de la gestión de recursos mediante técnicas como el uso del umbral adaptativo en la generación de eventos (Guinaldo et al., 2013). El estudio realizado en este trabajo enfocado en el movimiento de formación se extenderá al ámbito del desplazamiento de la formación en conjunto en tareas de navegación. Desde la perspectiva del grafo del SMR y las diferencias en el rendimiento entre agentes, se explorará el uso de grafos ponderados, que asignan un peso distinto a los enlaces entre agentes. De esta forma, se espera que la velocidad de convergencia a la formación pueda aumentar. Finalmente, para dar respuesta a la tendencia tecnológica actual, se afianzará el desarrollo de experiencias de realidad mixta y el uso de gemelos digitales en SMR.

Agradecimientos

Este trabajo ha sido realizado parcialmente gracias al apoyo de la Agencia Estatal de Investigación (AEI) a través de los proyectos PID2020-112658RB-I00/AEI/10.13039/501100011033, 2021V/-TAJOV/001, IEData 2016-6 y PID2022-139187OB-I00.

Referencias

Amsters, R., Slaets, P., 2020. Turtlebot 3 as a robotics education platform. In: Robotics in Education. Springer, Cham, Switzerland, pp. 170–181.
DOI: 10.1007/978-3-030-26945-6_16

Anderson, B. D., Yu, C., Fidan, B., Hendrickx, J. M., 2008. Rigid graph control architectures for autonomous formations. IEEE Control Systems Magazine 28 (6), 48–63.
DOI: 10.1109/MCS.2008.929280

Aranda, M., López-Nicolás, G., Sagüés, C., Mezouar, Y., 2015. Formation control of mobile robots using multiple aerial cameras. IEEE Transactions on Robotics 31 (4), 1064–1071.
DOI: 10.1109/TRO.2015.2452777

Aranda-Escolastico, E., Guinaldo, M., Heradio, R., Chacon, J., Vargas, H., Sánchez, J., Dormido, S., 2020. Event-based control: A bibliometric analysis of twenty years of research. IEEE Access 8, 47188–47208.
DOI: 10.1109/ACCESS.2020.2978174

Bansal, S., Akametalu, A. K., Jiang, F. J., Laine, F., Tomlin, C. J., 2016. Learning quadrotor dynamics using neural network for flight control. In: 2016 IEEE 55th Conference on Decision and Control (CDC). IEEE, Las Vegas, NV, USA, pp. 4653–4660.
DOI: 10.1109/CDC.2016.7798978

Barra, J., Scorletti, G., Lesecq, S., Zarudniev, M., Blanco, E., 2020. Attraction domain estimation of linear controllers for the attitude control of vtol vehicles: P/pi control of a quadrotor. In: 2020 European Control Conference (ECC). IEEE, St. Petersburg, Russia, pp. 1644–1649.
DOI: 10.23919/ECC51009.2020.9143612

Bogdan, P., Marculescu, R., 2011. Towards a science of cyber-physical systems design. In: 2011 IEEE/ACM Second international conference on cyber-physical systems, Chicago, IL, USA. IEEE, Chicago, IL, USA, pp. 99–108.
DOI: 10.1109/ICCPS.2011.14

Budaciu, C., Botezatu, N., Kloetzer, M., Burlacu, A., 2019. On the evaluation of the crazyflie modular quadcopter system. In: 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA). IEEE, Zaragoza, Spain, pp. 1189–1195.
DOI: 10.1109/ETFA.2019.8869202

Chee, K. Y., Jiahao, T. Z., Hsieh, M. A., 2022. Knode-mpc: A knowledge-based data-driven predictive control framework for aerial robots. IEEE Robotics and Automation Letters 7 (2), 2819–2826.

Cornejo, J., Magallanes, J., Denegri, E., Canahuire, R., 2018. Trajectory tracking control of a differential wheeled mobile robot: a polar coordinates control and lqr comparison. In: 2018 IEEE XXV International Conference on Electronics, Electrical Engineering and Computing (INTERCON). IEEE, Lima, Peru, pp. 1–4.
DOI: 10.1109/INTERCON.2018.8526366

Cortés, J., Egerstedt, M., 2017. Coordinated control of multi-robot systems: A survey. SICE Journal of Control, Measurement, and System Integration 10 (6), 495–503.
DOI: 10.9746/jcmsi.10.495

Didier, A., Parsi, A., Coulson, J., Smith, R. S., 2021. Robust adaptive model predictive control of quadrotors. In: 2021 European Control Conference (ECC). IEEE, Delft, Netherlands, pp. 657–662.
DOI: 10.23919/ECC54610.2021.9654893

Fidan, B., Yu, C., Anderson, B. D., 2007. Acquiring and maintaining persistence of autonomous multi-vehicle formations. IET Control Theory & Applications 1 (2), 452–460.
DOI: 10.1049/iet-cta:20050409

Garcia, G. A., Kim, A. R., Jackson, E., Keshmiri, S. S., Shukla, D., 2017. Modeling and flight control of a commercial nano quadrotor. In: 2017 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, Miami, FL, USA, pp. 524–532.
DOI: 10.1109/ICUAS.2017.7991439

Giernacki, W., Rao, J., Sladic, S., Bondyra, A., Retinger, M., Espinoza-Fraire, T., 2022. Dji tello quadrotor as a platform for research and education in mobile robotics and control engineering. In: 2022 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, Dubrovnik, Croatia, pp. 735–744.
DOI: 10.1109/ICUAS54217.2022.9836168

Giernacki, W., Skwierczyński, M., Witwicki, W., Wroński, P., Kozierski, P., 2017. Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering. In: 2017 22nd International Conference on Methods and Models in Automation and Robotics (MMAR). IEEE, Miedzyzdroje, Poland, pp. 37–42.
DOI: 10.1109/MMAR.2017.8046794

Grasso, P., Innocente, M. S., Tai, J. J., Haas, O., Dizqah, A. M., 2022. Analysis and accuracy improvement of uwb-tdoa-based indoor positioning system. Sensors 22 (23), 9136.
DOI: 10.3390/s22239136

Guinaldo, M., Dimarogonas, D. V., Johansson, K. H., Sánchez, J., Dormido, S., 2013. Distributed event-based control strategies for interconnected linear systems. IET Control Theory & Applications 7 (6), 877–886.
DOI: 10.1049/iet-cta.2012.0525

Guinaldo, M., Sánchez, J., Dormido, S., 2017. Control en red basado en eventos: de lo centralizado a lo distribuido. Revista Iberoamericana de Automática e Informática Industrial 14 (1), 16–30.
DOI: 10.1016/j.riai.2016.09.007

Hasseni, S.-E.-I., Abdou, L., Glida, H.-E., 2021. Parameters tuning of a quadrotor pid controllers by using nature-inspired algorithms. Evolutionary Intelligence 14, 61–73.
DOI: 10.1007/s12065-019-00312-8

Heemels, W. P., Johansson, K. H., Tabuada, P., 2012. An introduction to event-triggered and self-triggered control. In: 2012 IEEE 51st IEEE Conference on Decision and Control (CDC). IEEE, Maui, HI, USA, pp. 3270–3285.
DOI: 10.1109/CDC.2012.6425820

Heikkinen, J., Minav, T., Stotckaia, A. D., 2017. Self-tuning parameter fuzzy pid controller for autonomous differential drive mobile robot. In: 2017 XX

IEEE international conference on soft computing and measurements (SCM). IEEE, St. Petersburg, Russia, pp. 382–385.
DOI: 10.1109/SCM.2017.7970592

Krick, L., Broucke, M. E., Francis, B. A., 2009. Stabilisation of infinitesimally rigid formations of multi-robot networks. International Journal of Control 82 (3), 423–439.
DOI: 10.1080/00207170802108441

Lamraoui, H. C., Qidan, Z., Benrabah, A., 2017. Dynamic velocity tracking control of differential-drive mobile robot based on ladrc. In: 2017 IEEE International Conference on Real-time Computing and Robotics (RCAR). IEEE, Okinawa, Japan, pp. 633–638.
DOI: 10.1109/RCAR.2017.8311934

Leonard, N. E., Paley, D. A., Lekien, F., Sepulchre, R., Fratantoni, D. M., Davis, R. E., 2007. Collective motion, sensor networks, and ocean sampling. Proceedings of the IEEE 95 (1), 48–74.
DOI: 10.1109/JPROC.2006.887295

Mañas-Álvarez, F. J., Guinaldo, M., Dormido, R., Dormido-Canto, S., 2023. Scalability of cyber-physical systems with real and virtual robots in ros 2. Sensors 23 (13), 6073.
DOI: 10.3390/s23136073

Mañas-Álvarez, F.-J., Guinaldo, M., Dormido, R., Socas, R., Dormido, S., 2022. Formation by consensus in heterogeneous robotic swarms with twins-in-the-loop. In: ROBOT2022: Fifth Iberian Robotics Conference. Springer, Cham, Switzerland, pp. 435–447.
DOI: 10.1007/978-3-031-21065-5_36

Mañas-Álvarez, F.-J., Guinaldo, M., Dormido, R., Dormido, S., 2023. Robotic park: Multi-agent platform for teaching control and robotics. IEEE Access 11, 34899–34911.
DOI: 10.1109/ACCESS.2023.3264508

Nguyen, N. P., Lee, B. H., Xuan-Mung, N., Ha, L. N. N. T., Jeong, H. S., Lee, S. T., Hong, S. K., 2022. Persistent charging system for crazyflie platform. Drones 6 (8), 212.
DOI: 10.3390/drones6080212

Oh, K.-K., Park, M.-C., Ahn, H.-S., 2015. A survey of multi-agent formation control. Automatica 53, 424–440.
DOI: 10.1016/j.automatica.2014.10.022

Pichierri, L., Testa, A., Notarstefano, G., 2023. Crazychoir: Flying swarms of crazyflie quadrotors in ros 2. IEEE Robotics and Automation Letters 8 (8), 4713–4720.
DOI: 10.1109/LRA.2023.3286814

Ren, W., Sorensen, N., 2008. Distributed coordination architecture for multi-robot formation control. Robotics and Autonomous Systems 56 (4), 324–333.
DOI: 10.1016/j.robot.2007.08.005

Silano, G., Aucone, E., Iannelli, L., 2018. Crazys: a software-in-the-loop platform for the crazyflie 2.0 nano-quadcopter. In: 2018 26th Mediterranean Conference on Control and Automation (MED). IEEE, Zadar, Croatia, pp. 1–6.
DOI: 10.1109/MED.2018.8442759

Soares, J. M., Navarro, I., Martinoli, A., 2016. The khepera iv mobile robot: performance evaluation, sensory data and software toolbox. In: Robot 2015: Second Iberian Robotics Conference: Advances in Robotics, Volume 1. Springer, Cham, Switzerland, pp. 767–781.
DOI: 10.1007/978-3-319-27146-0_59

Štefek, A., Pham, V. T., Krivanek, V., Pham, K. L., 2021. Optimization of fuzzy logic controller used for a differential drive wheeled mobile robot. Applied Sciences 11 (13), 6023.
DOI: 10.3390/app11136023

Taffanel, A., Rousselot, B., Danielsson, J., McGuire, K., Richardsson, K., Eliasson, M., Antonsson, T., Hönig, W., 2021. Lighthouse positioning system: Dataset, accuracy, and precision for uav research. In: ICRA Workshop on Robot Swarms in the Real World. ArXiv.
DOI: 10.48550/arXiv.2104.11523

Vázquez, U., González-Sierra, J., Fernández-Anaya, G., Hernández-Martínez, E. G., 2021. Análisis del desempeño de un control pid de orden fraccional en un robot móvil diferencial. Revista Iberoamericana de Automática e Informática Industrial 19 (1), 74–83.
DOI: 10.4995/riai.2021.15036

Vicon Motion Systems, 2022. Accuracy in motion. [Consulta el 01-07-2023].
URL: https://www.vicon.com/wp-content/uploads/2022/07/Vicon-Metrology-Solutions.pdf

Vágner, M., Palkovics, D., Kovács, L., 2022. 3d localization and data quality estimation with marvelmind. In: 2022 IEEE 2nd Conference on Information Technology and Data Science (CITDS). IEEE, Debrecen, Hungary, pp. 302–307.
DOI: 10.1109/CITDS54976.2022.9914386

Zekry, O. H., Attia, T., Hafez, A. T., Ashry, M., 2023. Pid trajectory tracking control of crazyflie nanoquadcopter based on genetic algorithm. In: 2023 IEEE Aerospace Conference. IEEE, Big Sky, MT, USA, pp. 1–8.
DOI: 10.1109/AERO55745.2023.10115538