

Jornadas de Automática

Un benchmark para el diseño y control de formaciones en sistemas multi-agente

Mañas-Álvarez, F.J.^{a,*}, Guinaldo, M.^a, Dormido, R.^a, Dormido, S.^a

^aDepartamento de Informática y Automática, E.T.S.I. Informática, Universidad Nacional de Educación a Distancia (UNED),
C/ Juan del Rosal, n° 16, 28040, Madrid, España.

To cite this article: Mañas-Álvarez, F.J., Guinaldo, M., Dormido, R., Dormido, S. 2024. A benchmark for the design and control of formations in multi-agent systems. *Jornadas de Automática*, 45. <https://doi.org/10.17979/ja-cea.2024.45.10749>

Resumen

La investigación en Sistemas Multi-Agente Robóticos (SMAR) ha recibido gran atención en los últimos años, ya que la cooperación entre agentes permite al sistema realizar tareas complejas a partir de comportamientos individuales relativamente sencillos. Uno de los principales retos en este campo es lograr y mantener la formación del sistema multiagente, es decir, coordinar múltiples agentes para satisfacer las restricciones impuestas en sus estados. El presente trabajo pretende abordar este reto en el marco de Robotic Park, una plataforma experimental integrada en una red ROS2, que admite experiencias con SMAR en un entorno virtual, en un entorno físico o en un esquema híbrido, utilizando vehículos aéreos y terrestres. El benchmark propuesto permite tanto la generación de una formación objetivo como el diseño de los algoritmos de control para lograrla. Además, posibilita la evaluación del rendimiento del sistema desde diferentes aspectos, como el rendimiento del controlador local, el número de mensajes de comunicación o el uso de la CPU.

Palabras clave: Sistema Multi-Agente, Control de Coordinación, UAV, Robots, Realidad Virtual

A benchmark for the design and control of formations in multi-agent systems

Abstract

Research in cooperative multi-agent robotic systems (MARS) has received wide attention in recent years, since the cooperation between agents allows the system to perform complex tasks from relatively simple individual behaviours. One of the main challenges in this field is to achieve and maintain the formation of the multi-agent system, i.e., coordinating multiple agents to achieve prescribed constraints on their states. This work aims to address this challenge within the framework of Robotic Park, an experimental platform integrated into a ROS2 network, which supports MARS experiences in a virtual environment, in a physical environment or in a hybrid scheme, using aerial and ground vehicles. The proposed benchmark allows both the generation of a target formation and the design of the control algorithms to achieve it. In addition, it enables the system performance evaluation from different aspects, such as the performance of the local controller, the number of communication messages, or CPU usage.

Keywords: Multi-Agent System, Coordination control, UAV, Robots, Virtual Reality

1. Introducción

El uso de sistemas robóticos autónomos que trabajan de forma cooperativa para diferentes tareas relacionadas con la robótica ha ido en aumento en los últimos años. En este contexto, cada entidad del sistema se denomina *agente*, y el sistema se denomina *sistema multi-agente robótico* (SMAR). En

particular, el despliegue de una formación para completar tareas específicas de forma segura y eficiente es uno de los principales intereses, como en los casos de monitorización o vigilancia autónomas (Leonard et al., 2007). En estas tareas mantener la formación de los robots juega un papel crucial, y el diseño de leyes de control distribuido que garanticen la

*Autor para correspondencia: fjmanas@dia.uned.es
Attribution-NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA 4.0)

consecución y mantenimiento de dicho objetivo es una línea de investigación activa (Lawton et al., 2003).

Hay diferentes formas de caracterizar una formación que conducen a diferentes estrategias de control de la formación. Si nos centramos en esta caracterización en términos de la capacidad de detección y la topología de los agentes, la pregunta natural es qué variables son detectadas y qué variables son controladas por los sistemas multi-agente para lograr la formación. Atendiendo a esta cuestión, se puede establecer la siguiente clasificación (Oh et al., 2015):

- Control basado en la posición. Los agentes pueden percibir sus propias posiciones respecto de un sistema de coordenadas global y pueden controlarlas activamente para lograr la formación deseada, de modo que los agentes puedan moverse sin interactuar entre sí.
- Control basado en el desplazamiento. Los agentes pueden detectar las posiciones relativas de sus vecinos y controlar su posición para lograr la formación deseada. Al definirse en términos de desplazamientos deseados respecto a un sistema de coordenadas global, se requiere la orientación de los sistemas de coordenadas globales.
- Control basado en la distancia. Los agentes miden las posiciones relativas respecto a sus vecinos, pero las variables que se controlan son las distancias entre agentes. Por tanto, la orientación de los sistemas de coordenadas locales no está necesariamente alineada entre sí. Sin embargo, las leyes de control existentes no son lineales, lo que dificulta el análisis.

La interacción entre agentes suele modelarse en términos de un grafo. A pesar de la direccionalidad de los vínculos entre agentes, las interconexiones de los grafos pueden tener distintos significados y esto depende generalmente de las capacidades de sensorización. Por ejemplo, si los agentes están equipados con sensores que permiten la medición de las variables que necesitan ser detectadas, entonces la comunicación podría no ser necesaria. Sin embargo, si estas variables no pueden ser accedidas directamente por los sensores, entonces los agentes necesitarían transmitir las medidas a otros nodos de la red, y entonces la topología se interpreta como un grafo de comunicación (Pantelimon et al., 2019). En este caso, los aspectos que conciernen a la comunicación, como el protocolo o la estructura de la red, pueden afectar al rendimiento del control, por lo que es una cuestión que debe tratarse con cuidado. Además, la frecuencia de la comunicación tiene un impacto directo en el rendimiento del control, pero también en la duración de la batería de los dispositivos autónomos.

Cuando el número de agentes del sistema es grande se denomina enjambre. Un reto importante en el control en estos sistemas es el desarrollo de estructuras de control escalables, es decir, que no incrementen su complejidad con el número de agentes. Para ello, una de las soluciones más frecuente es emplear estructuras descentralizadas que ejecutan el control a bordo de los agentes. Sin embargo, en algunos dispositivos la capacidad de cálculo es limitada y, a veces, las tareas requieren una unidad central, por ejemplo, en sistemas externos de posicionamiento en interiores. Además, cuando el sistema incluye

agentes virtuales o Gemelos Digitales (GD), entonces la preocupación está en los recursos computacionales requeridos por las herramientas de simulación. Por lo tanto, la escalabilidad y el estudio del impacto del aumento del número de agentes en el control es un aspecto de gran interés.

A pesar del interés que los SMAR han generado tanto en la industria como en el mundo académico por su potencial e idoneidad en escenarios reales, todavía faltan herramientas adecuadas para validar, evaluar y comparar diferentes enfoques de forma exhaustiva. Aunque se pueden encontrar varios benchmarks en la literatura sobre SMAR (Yan et al., 2015; Portugal et al., 2019), ninguno de ellos, según el conocimiento de los autores, aborda el problema del control de la formación. Por lo tanto, consideramos importante para la comunidad investigadora en ingeniería de control diseñar un benchmark con este fin que permita evaluar los diferentes retos mencionados que surgen en la tarea de formación en SMAR.

Robotic Park (RP) (Mañas-Álvarez et al., 2023b) es la plataforma experimental empleada en el benchmark propuesto. Se trata de una plataforma de interior heterogénea, flexible y fácil de usar desarrollada recientemente para realizar experimentos SMAR en el Departamento de Informática y Automática de la UNED. Permite llevar a cabo experiencias virtuales, reales o híbridas combinando agentes físicos (y sus GD) y virtuales en experiencias de Realidad Mixta (RM). Esto es posible gracias a la integración de todos los agentes en una red de ROS 2, la cual hace indistinguibles la naturaleza de los agentes para sus vecinos y permite la comunicación en tiempo real entre los agentes físicos, sus gemelos, y los agentes virtuales.

Por consiguiente, el objetivo de este benchmark es doble. Por un lado, la generación de la formación objetivo basada en un conjunto de especificaciones dadas. Por otro lado, el diseño de algoritmos de control para lograr la formación. La evaluación de las soluciones propuestas tiene en cuenta la complejidad de los algoritmos, el rendimiento del controlador de coordinación, el número de mensajes de comunicación entre agentes o el uso de la CPU.

Este trabajo se estructura de la siguiente manera. En la sección 2 se describen las principales características del problema estudiado (plataforma experimental, objetivos y control de formación). En la sección 3 se presentan los dos problemas abordados en el benchmark, así como su diseño, la configuración experimental de los casos propuestos y las métricas de rendimiento utilizadas para evaluar el sistema de control de la formación. La sección 4 describe el acceso al benchmark y los recursos disponibles, detallando paso a paso cómo llevar a cabo una validación virtual. Por último, en la sección 5 se discuten los resultados y se plantean las líneas futuras de trabajo.

2. Descripción del problema

2.1. Plataforma Experimental

Robotic Park (Mañas Álvarez, 2023) es una plataforma experimental diseñada para respaldar la investigación y la enseñanza para respaldar el trabajo en el ámbito de los sistemas ciberfísicos aplicados a la robótica. Su propósito abarca desde la investigación en SMAR hasta la docencia a distancia en

trabajos fin de grado y máster. Este enfoque presenta desafíos significativos, como la necesidad de flexibilidad en el diseño de experiencias, una infraestructura robusta para la tolerancia a fallos, y un mantenimiento constante para garantizar su vanguardia y funcionalidad.

La arquitectura de la plataforma se ha concebido bajo dos principios fundamentales: elasticidad y flexibilidad. La elasticidad permite conmutar entre una configuración centralizada o distribuida, tanto en hardware como en software. Esto se consigue eligiendo qué componentes se ejecutan en cada unidad computacional (ya sean ordenadores, microcontroladores o sistemas embebidos). Por otro lado, la flexibilidad de la arquitectura se logra estableciendo un conjunto de “reglas” que todos los componentes del sistema deben cumplir, garantizando así una comunicación sin conflictos entre elementos, independientemente de su entorno o naturaleza. Esto es especialmente valioso para que la plataforma pueda reproducir experiencias virtuales en entornos reales sin necesidad de modificar los controladores de los agentes, o el intercambio de módulos diseñados por los propios usuarios con los disponibles por defecto. Esto es posible gracias al uso de ROS 2 como middleware que permite abstraer la heterogeneidad existente a nivel hardware y software y posibilita el funcionamiento en tiempo real en las experiencias de RM.

La heterogeneidad característica de RP se muestra parcialmente en la tipología de los robots que la componen. Está compuesta por robots móviles y aéreos (ver Figura 1). Los agentes aéreos incluyen los cuadricópteros Crazyflie 2.1, reconocidos por su diseño de código abierto y su sensorización modular adaptable. Se emplean igualmente los DJI Tello, plataformas comerciales con una autonomía de vuelo superior y cámaras frontales integradas con mayor resolución. En cuanto a los agentes móviles, destacan los Khepera IV, una plataforma comercial con una sensorización precisa y compacta, y los Turtlebot3, populares en el ámbito de la robótica y ROS por ser de código abierto y por su amplio uso en investigación y docencia.



Figura 1: Agentes disponibles en Robotic Park

En RP, las herramientas de simulación desempeñan un papel crucial. Proporcionan un entorno seguro para que los usuarios desarrollen sus habilidades y validen sus algoritmos antes de trabajar en un entorno físico. Además, son esenciales para crear experiencias de RM y el uso de GD. La RM permite implementar sistemas con un mayor número de agentes de lo que sería posible en un entorno físico debido a limitaciones de espacio. El uso de GD en esta plataforma se ha enfocado en dos aplicaciones fundamentales. En primer lugar, respecto

a la sensorización permite mejorar la capacidad de los agentes físicos con sensores más complejos si el entorno virtual es una réplica precisa del real y percibir aquellos agentes que sean exclusivamente virtuales y viceversa. Por otro lado, suponen un agente de respaldo en caso de fallo crítico del agente real para evitar la inestabilidad en los agentes enlazados en experiencias de formación.

2.2. Formación objetivo

Se ha considerado que la formación objetivo del SMAR sea una superficie semiesférica. Dicha superficie permite una mejor cobertura del área en comparación con otras formas geométricas, siendo ideal en aplicaciones como cartografía, vigilancia o fotografía aérea. Un SMAR puede modelarse como un conjunto \mathcal{V} compuesto por N agentes. El primer objetivo es conseguir una topología de formación en el espacio 3D que cumpla las siguientes restricciones:

1. Todos los agentes deben mantener la misma distancia R respecto a un punto de referencia $p_0 = (0 \ 0 \ z_0)^\top$, $z_0 \geq 0$.
2. En la posición final, para todo agente $i \in \mathcal{V}$, la posición $p_i = (p_{i,x} \ p_{i,y} \ p_{i,z})^\top$ debe cumplir $p_{i,z} \geq z_0$.

Estas dos condiciones generan un subespacio para el despliegue de la formación que es la superficie de una semiesfera de radio R y centro p_0 :

$$x^2 + y^2 + (z - z_0)^2 = R^2 \quad (1)$$

La tercera restricción de la formación impuesta es:

3. Los agentes deben distribuirse por la superficie virtual de la semiesfera de forma que se optimice su cobertura.

Este último requisito conecta con un problema clásico de optimización, el problema de Thomson (Thomson, 1904). Este problema aparentemente simple en su formulación es uno de los problemas matemáticos abiertos debido a la complejidad de la solución general, y la computabilidad o manejabilidad de algunos casos simples.

Un SMAR se puede modelar como un grafo estático no dirigido \mathcal{G} , descrito mediante el conjunto de nodos-agente \mathcal{V} y el conjunto de aristas \mathcal{E} . Para cada agente i , el conjunto de sus vecinos \mathcal{N}_i es aquel que cumple $\mathcal{N}_i = \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$. Nótese que $|\mathcal{N}_i| = \deg v_i$, donde $|\cdot|$ representa la cardinalidad del conjunto \mathcal{N}_i y \deg es el grado del vértice v_i asociado al nodo i .

La solución propuesta para esta parte del benchmark debe generar dos conjuntos:

- El conjunto de aristas \mathcal{E} para la formación objetivo, que implícitamente proporciona el vecindario \mathcal{N}_i para cada agente i , de forma que la formación resultante sea factible y se minimicen las ambigüedades.
- El conjunto de distancias euclídeas objetivo, denotado como d_{ij}^* , para todo par $(i, j) \in \mathcal{E}$.

Finalmente, el primer problema a resolver puede enunciarse como:

Problema 1. Dado un SMAR cuya topología se modela mediante un grafo $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, encontrar el conjunto de aristas \mathcal{E} y el correspondiente conjunto de distancias objetivo para el control de la formación, $\{d_{ij}^* : (i, j) \in \mathcal{E}\}$, tal que la cobertura de la superficie (1) se aproxime a la solución óptima.

2.3. Control de formación

La arquitectura de control considerada para este sistema sigue una estructura jerárquica distribuida (Mañas-Álvarez et al., 2023a). Los controladores de los niveles inferiores son PIDs responsables del posicionamiento individual de los agentes. El controlador de coordinación del nivel superior recibe las posiciones de los agentes y genera una referencia para el nivel inferior. Dado que los controladores de estabilización se diseñan e implementan por separado, esto permite considerar modelos simplificados de los agentes para el nivel de control superior, como modelos de integrador único para las posiciones de referencia:

$$\dot{p}_i(t) = u_i(t), \quad (2)$$

donde $u_i(t) = (u_{i,x}(t) \ u_{i,y}(t) \ u_{i,z}(t))^T \in \mathbb{R}^3$, es decir, se genera una señal de control para cada componente x , y , y z .

El objetivo de control se define en términos de un conjunto de distancias objetivo, d_{ij}^* , generadas como solución del Problema 1. El problema de control se resuelve si cada agente i alcanza la distancia objetivo d_{ij}^* , mientras que mantiene la distancia al punto de referencia p_0 en R . Así pues, el segundo problema de este benchmark puede enunciarse como:

Problem 2. *Supongamos que el controlador de posición para cada robot de un SMAR está dado de manera que el modelo (2) puede ser asumido para las posiciones de referencia. Hallar la ley de control distribuido $u_i(t)$ para el controlador de coordinación y para cada agente i*

$$u_i(t) = f_i(p_i, \{p_j - p_i, d_{ij}^*, j \in \mathcal{N}_i\}, p_0, R), \quad (3)$$

tal que para cada nodo vecino $j \in \mathcal{N}_i$, la distancia euclidiana entre ellos, $d_{ij} = \|p_i - p_j\|$, satisfaga

$$\lim_{t \rightarrow \infty} d_{ij}(t) = d_{ij}^*, \quad j \in \mathcal{N}_i, \quad (4)$$

así como

$$\lim_{t \rightarrow \infty} \|p_i(t) - p_0\| = R. \quad (5)$$

3. Benchmark

A continuación se proporcionan los detalles del benchmark para los problemas de definición de la formación objetivo y de control de la misma. Para ambos problemas, el número de agentes N se fija en 15 y el radio de la superficie semiesférica (1) es $R = 1$ [m].

3.1. Problema 1

El algoritmo para resolver el Problema 1 se proporcionará en un fichero Matlab (*.m). Los parámetros de entrada serán $N = 15$ y $R = 1$ y generará como salidas el conjunto de aristas \mathcal{E} que define la topología y el conjunto de distancias objetivo d_{ij}^* . Para evaluar el rendimiento de la solución propuesta se utilizan las siguientes métricas de rendimiento:

- *Tiempo de CPU* (τ_{CPU}). Tiempo necesario para ejecutar la solución propuesta. Supone una buena aproximación a la complejidad del algoritmo. Para calcularlo se utilizarán los comandos de Matlab `tic` y `toc`.

- *La discrepancia de distancia* ($\delta(d_{ij})$). Computa la diferencia entre el valor máximo y el mínimo del conjunto $\{d_{ij}^*\}$ como forma de medir la uniformidad en la distribución de nodos en la formación objetivo:

$$\delta(d_{ij}) = \max_{(i,j) \in \mathcal{E}} d_{ij}^* - \min_{(i,j) \in \mathcal{E}} d_{ij}^*. \quad (6)$$

- *Energía potencial de la malla* (E_d). Como variación del problema de Thompson (Tomson, 1904), sólo se considerará la distancia entre nodos conectados:

$$E_d = \sum_{(i,j) \in \mathcal{E}} \frac{1}{(d_{ij}^*)^2}. \quad (7)$$

Así, la función para evaluar la bondad de la solución propuesta al Problema 1 es:

$$J_1 = \lambda_1 \tau_{CPU} + \lambda_2 \delta(d_{ij}) + \lambda_3 E_d, \quad (8)$$

donde $\lambda_1 = 2,4283$, $\lambda_2 = 1,0$, y $\lambda_3 = 0,0079$. Los parámetros se han elegido de modo que cada índice de rendimiento contribuyera por igual al valor de J_1 en el caso de referencia.

3.2. Problema 2

En el problema de control de formación, se asume que la solución generada por el Problema 1 se emplea como entrada. En el estado inicial del sistema, los agentes se encuentran distribuidos de forma aleatoria en el plano $X - Y$ a una cota $z = 0,0$ [m] y se elevan hasta alcanzar un nuevo estado estacionario en la cota $z = 1,0$ [m] (véase la Figura 2(a)). La distribución por defecto está disponible en el archivo de configuración del experimento `config.yaml`. La Figura 2(b) muestra un ejemplo de formación final deseada dada como solución factible al Problema 1.

Para evaluar el rendimiento de la ley de control propuesta, se consideran los siguientes parámetros para cada agente i :

- Número de *mensajes publicados a la red*, $n_{m,i}$. Este parámetro evalúa el protocolo de comunicación definido entre los agentes.
- Número de *ejecuciones del controlador*, $n_{u,i}$.
- *Tiempo de establecimiento*, $t_{e,i}$. Tiempo necesario para alcanzar el error cero y permanecer dentro de 2% del error inicial para cada arista. Este término se calcula localmente en cada agente para mantener la arquitectura distribuida del SMAR.
- *Integral del Error Absoluto*, *IEA*. Representa el error global de formación ponderando todos los errores por igual a lo largo del tiempo. Estos errores corresponden a la desviación del valor de cada arista con respecto al valor deseado. Como el cálculo de los errores se hace localmente, el resultado final se divide por dos para evitar contar dos veces el error de cada arista. Se calcula de la siguiente manera:

$$IEA_i = \frac{1}{2} \sum_{k=1}^{N_k} \int_0^T \|d_{ik}^* - d_{ik}\| dt, \quad (9)$$

donde k representa los vecinos del agente i y N_k es el número de aristas.

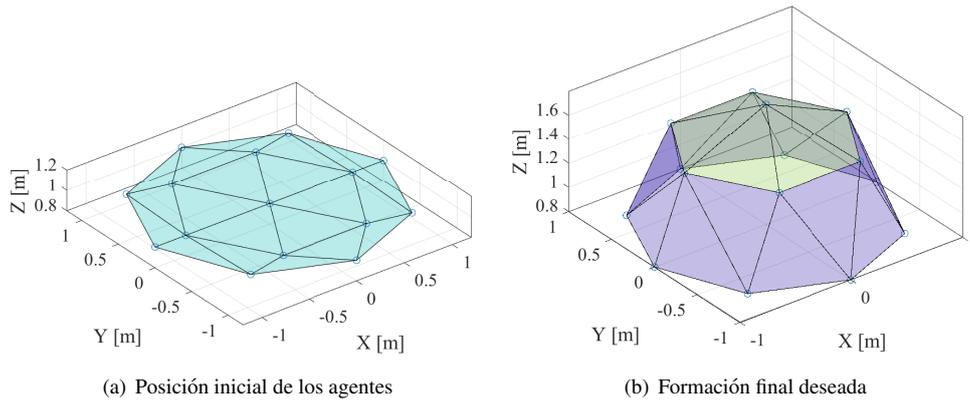


Figura 2: Geometrías por defecto para el benchmark propuesto

- *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. Los errores que persisten en el tiempo se ponderan mucho más que los que existen al principio de la respuesta. Como el cálculo del error se hace localmente, el resultado se divide por dos para evitar contar dos veces el error de cada arista. Se calcula de la siguiente manera:

$$IEAT_i = \frac{1}{2} \sum_{k=1}^{N_k} \int_0^T \|d_{ik}^* - d_{ik}\| t dt. \quad (10)$$

Por lo tanto, la función de coste para evaluar las soluciones del Problema 2 es

$$J_2 = \sum_{i=1}^N (\lambda_1 n_{m,i} + \lambda_2 n_{u,i} + \lambda_3 t_{e,i} + \lambda_4 IEA_i + \lambda_5 IEAT_i), \quad (11)$$

donde $\lambda_1 = 0,02$, $\lambda_2 = 0,1$, $\lambda_3 = 0,3$, $\lambda_4 = 0,1$, y $\lambda_5 = 0,01$. Al igual que en (8), el valor de los parámetros se han elegido de modo que cada índice de rendimiento contribuya por igual al valor de J_2 en el caso de referencia.

4. Accesibilidad

El benchmark propuesto está disponible para que la comunidad investigadora pueda comenzar a utilizarlo y poder valorar su impacto. Todo el código necesario para su ejecución se puede encontrar en un repositorio público de la organización de Github de RP¹. Para su correcto funcionamiento será necesario ejecutarlo en una máquina con Ubuntu 22.04 y tener instalado ROS 2 Humble. Para instalar RP en su máquina, se clona el repositorio y se ejecuta el archivo de instalación `install.sh`. Este fichero asegura la instalación de todos los paquetes necesarios y compila el proyecto en el espacio de trabajo.

La Figura 3 muestra una visión general de la arquitectura general del benchmark propuesto para la resolución de los Problemas 1 y 2 mediante un diagrama de flujo. A la izquierda se muestran los componentes relacionados con el Problema

1. En el repositorio mencionado anteriormente se proporciona un archivo de plantilla de función Matlab `problem1.m` con dos parámetros de entrada (N y R) y tres parámetros de salida: 1) E , una matriz $N_e \times 2$ donde cada fila denota una arista (i, j) y N_e es el número de aristas diseñadas; 2) D , un vector de longitud N_e donde el elemento k -th es la distancia objetivo para la arista en la fila k -th de E ; y 3) P , una matriz $N \times 3$ donde cada fila representa las coordenadas (x, y, z) de un agente.

La solución obtenida, a través del fichero `RP_benchmark.m`, valida los resultados obtenidos, evalúa el Problema 1 y genera la topología SMAR necesaria para abordar el Problema 2. Esta topología supone la entrada a la aplicación de diseño experimental que genera los ficheros necesarios para la ejecución de RP. En esta etapa, se pueden modificar las características de comunicaciones y controladores para los agentes del Problema 2 (`Config.yaml`). En el centro de la Figura 3, se muestra la simulación (Webots) y los componentes, en forma de nodos dentro del dominio ROS 2, ejecutados en el benchmark. Un archivo de lanzamiento de ROS 2 escrito en Python (`experience.launch.py`) está listo para ejecutar todos los nodos necesarios para resolver el Problema 2. A la derecha, se muestran los elementos responsables de registrar, validar, y evaluar los resultados del Problema 2.

5. Conclusiones

El interés creciente por los Sistemas de Multi-Agente Robóticos en diferentes aplicaciones y retos del mundo real estriba en las mejoras de rendimiento que pueden ofrecer frente a sistemas individuales. Hay en la literatura soluciones con SMAR basadas en drones para obtener información sobre algún área de interés. En el Problema 1 del benchmark presentado se ha considerado un despliegue de los agentes sobre una superficie virtual. La solución planteada integra herramientas que permiten definir una superficie de formación de objetivos de forma que los agentes se distribuyan sobre una semiesfera. Aunque se han presentado diferentes aproximaciones para abordar el problema de la distribución uniforme de puntos (Hardin et al., 2016), estos trabajos no se preocupan por la

¹<https://github.com/Robotic-Park-Lab/RoboticPark>

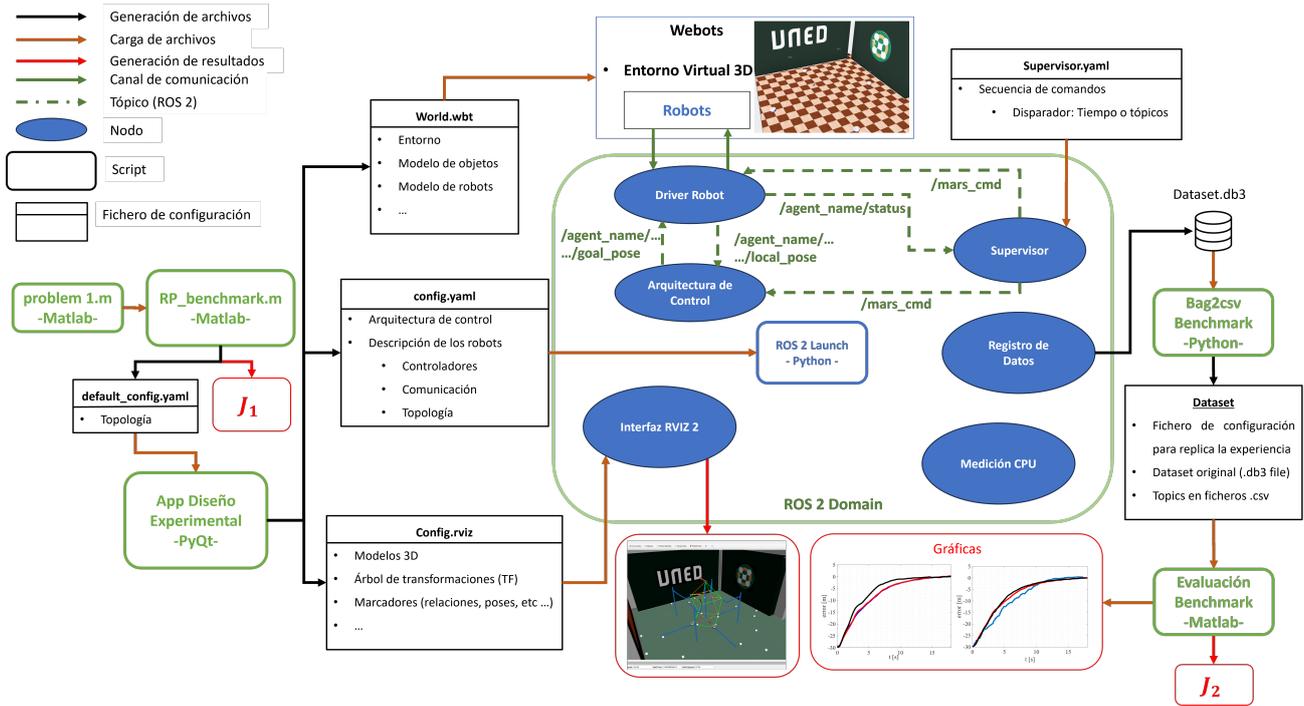


Figura 3: Diagrama de flujo del benchmark propuesto en Robotic Park

generación de la topología que es un aspecto crucial en este benchmark.

Además, lograr la formación deseada con un movimiento óptimo de los agentes se ha convertido en todo un reto. Para ello, es esencial desarrollar y diseñar nuevos algoritmos de control para lograr la formación de manera eficiente y robusta, tanto a nivel individual como colectivo. Esta es la motivación del Problema 2, ya que considera el diseño y evaluación de un controlador de formación para lograr un despliegue óptimo sobre la superficie definida en el Problema 1. Por último, es importante destacar que el código y la plataforma proporcionados en este benchmark están preparados para llevar a cabo diseños más complejos y preguntas de investigación adicionales para SMAR.

Como trabajo futuro, se plantean varias líneas que permitirán aprovechar la flexibilidad de Robotic Park. Algunas de estas ideas son la realización de experiencias en modo híbrido que permitiría teleoperar los robots reales en RP de forma remota asegurando la estabilidad en caso de pérdida de conexión, añadir heterogeneidad a las experiencias utilizando robots con diferentes restricciones y evaluar diferentes algoritmos y arquitecturas de control, tanto a nivel individual como de coordinación.

Agradecimientos

Este trabajo fue financiado en parte por la Agencia Estatal de Investigación (AEI) a través de los proyectos PID2020-112658RB-I00/AEI/10.13039/501100011033, IE-Data 2016-6 y PID2022-139187OB-I00.

Referencias

Hardin, D. P., Michaels, T., Saff, E. B., 2016. A comparison of popular point configurations on S^2 . Dolomites Research Notes on Approximation. DOI: 10.48550/arXiv.1607.04590

Lawton, J. R., Beard, R. W., Young, B. J., 2003. A decentralized approach to formation maneuvers. IEEE transactions on robotics and automation 19 (6), 933–941. DOI: 10.1109/TRA.2003.819598

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/JPR0C.2006.887295

Mañas Álvarez, F. J., 2023. Development, control and evaluation of a heterogeneous multi-agent robotic platform. Ph.D. thesis, UNED. Universidad Nacional de Educación a Distancia.

Mañas-Álvarez, F.-J., Guinaldo, M., Dormido, R., Dormido, S., 2023a. Muestreo y comunicación: impacto en el control de formaciones en sistemas multi-robot heterogéneos. Revista Iberoamericana de Automática e Informática industrial 21 (2), 125–136. DOI: 10.4995/riai.2023.20155

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

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

Pantelimon, G., Tepe, K., Carrievau, R., Ahmed, S., 2019. Survey of multi-agent communication strategies for information exchange and mission control of drone deployments. Journal of Intelligent & Robotic Systems 95, 779–788. DOI: 10.1007/s10846-018-0812-x

Portugal, D., Iocchi, L., Farinelli, A., 2019. A ros-based framework for simulation and benchmarking of multi-robot patrolling algorithms. In: Koubaa, A. (Ed.), Robot Operating System (ROS) The Complete Reference (Volume 3). Springer, Cham, Switzerland, pp. 3–28. DOI: 10.1007/978-3-319-91590-6_1

Tomson, J., 1904. On the structure of the atom: an investigation of the stability and periods of oscilation of a number of corpuscles arranged at equal intervals around the circumference of a circle; with application of the results to the theory atomic structure. Philos. Mag. Series 6 7 (39), 237.

Yan, Z., Fabresse, L., Laval, J., Bouraqadi, N., 2015. Metrics for performance benchmarking of multi-robot exploration. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, Hamburg, Germany, pp. 3407–3414. DOI: 10.1109/IROS.2015.7353852