IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 23 March 2023, accepted 31 March 2023, date of publication 5 April 2023, date of current version 12 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3264508

RESEARCH ARTICLE

Robotic Park: Multi-Agent Platform for Teaching Control and Robotics

FRANCISCO-JOSÉ MAÑAS-ÁLVAREZ, MARÍA GUINALDO, RAQUEL DORMIDO, AND SEBASTIÁN DORMIDO, (Member, IEEE)

Department of Computer Sciences and Automatic Control, Universidad Nacional de Educación a Distancia (UNED), 28040 Madrid, Spain

Corresponding author: Francisco-José Mañas-Álvarez (fjmanas@dia.uned.es)

This work was supported in part by the Agencia Estatal de Investigación (AEI) under Project PID2020-112658RB-I00/AEI/10.13039/501100011033, Project 2021V/-TAJOV/001, and Project IEData 2016-6.

ABSTRACT Multi-agent system research is a hot topic in different application domains. In robotics, multi-agent robot systems (MRS) can realize complex tasks even if the behavior of each agent seems simple thanks to the cooperation between them. Although many control algorithms for MRS are proposed, few experimental results are validated on real data, being essential to building new testbeds to conduct MRS research and teaching. Moreover, most existing platforms for experimentation do not offer an overall solution allowing software and hardware design tools. This paper describes the design and operation of Robotic Park, a new indoor experimental platform for research in MRS. The heterogeneity and flexibility of its configuration are two of its main contributions. It supports control design and validation of MRS algorithms. Experiences can be carried out in a virtual environment, physical environment, or under a hybrid scheme, as digital twins have been developed in Gazebo and Webots. Currently, two types of aerial vehicles (Crazyflie 2.X and DJI Tello) and two types of differential mobile robots (Turtlebot3 and Khepera IV) are available. Both internal and external positioning systems using different technologies such as Motion Capture or Ultra-WideBand are also available for experiences. All components are connected through ROS2 (Robot Operating System 2) which enables experiences under a centralized, distributed, or hybrid scheme, and different communication strategies can be implemented. All these features are novelties with respect to other existing platforms. A mixed reality experience that addresses the problem of formation control using event-based control illustrates the platform usage.

INDEX TERMS Motion control, multi-agent systems, robotics education.

I. INTRODUCTION

ROBotics is an inherently interdisciplinary field involving different domains such as engineering, computer science, or control [1]. The correct synergy between these fields entails a more efficient performance of the developed systems. For instance, the quality of sensors' measurements will have a direct impact on the control performance, and some tasks can be unfeasible if minimum accuracies are not reached [2]. These issues are more challenging when the system is composed of more than one robot and they have to cooperate. Multi-Robot Systems (MRSs) [3], also known as swarms when the number of agents is high [4], is one of the areas with great interest in robotics. These systems are designed to solve tasks that are computationally complex or highly time-consuming for a single robot. A good configuration of MRS directly influences an increase in efficiency, effectiveness, flexibility, and fault tolerance [5].

MRS is an extensive research field, and therefore, the taxonomy criteria of these systems are varied: the diversity of its agents (homogeneous or heterogeneous) [6], its control and communication architecture (centralized, distributed or hybrid) [7], its operating environment (indoors or outdoors) [8], the positioning system, the technology used in communication such as Ultra-Wide Band (UWB) [9], the technique used to build the environment [10], among others. Additionally, the variety of control algorithms available in the literature for MRS is wide. In this way, for formation

The associate editor coordinating the review of this manuscript and approving it for publication was Shihong Ding.

problems by consensus or leader-follower (one of the most studied for MRS), it is possible to find studies focused on the most traditional control algorithms such as Proportional Integral Derivative (PID) controllers [11] or Linear-Quadratic Regulator (LQR) [12] or in more complex algorithms such as robust control [13], predictive control [14], intelligent control through neural networks [15] or genetic algorithms [16].

Experimental platforms, or testbeds, are essential tools to evaluate and validate in-depth the developments of MRS in a controlled environment. Depending on their nature, these platforms can be virtual, physical or hybrid [17]. On the one hand, full virtual platforms are the most accessible tool to develop and validate algorithms since they do not require a reserved physical space and their only constraint is the computing capacity of the system on which they run. The most used software tools are Gazebo, Webots, CoppeliaSim and CARLA [18]. On the other hand, the physical environments allow validating in a real platform the MRS theoretical developments. Some examples of these platforms are Robotarium [19], the first MRS platform remotely accessible or DuckieTown [20], Pi-puck ecosystem [21] designed for differential vehicles. Other low-cost and open-source mobile robot homogeneous platforms can be found in SMARTmBOT [22] and HeRo [23]. In this way, the main operating experimental platforms for MRS make use of differential action mobile robots, being very few heterogeneous platforms. Finally, hybrid platforms are the so-called mixed reality, combining real and virtual robots [24]. Within this category are the Digital Twins (DTs), which are virtual replicas of real physical systems that, unlike conventional simulators or digital models, can interact with the physical system in real-time through bidirectional communication [25].

Although virtual platforms allow working with any system in almost any scenario in a faster and safer way than using real robots, it often happens that algorithms with perfect simulation results do not work under real-world conditions. Indeed, most of the results for distributed MRS are only still being validated in simulation, being crucial to building new testbeds to conduct MRS research.

This paper describes the development of a complete heterogeneous experimental platform called *Robotic Park*. This platform is designed to support MRS experiences combining different types of vehicles: micro-aerial quadcopter Crazyflie 2.1 [26] and DJI Tello [27], and differential mobile robots such as Turtlebot3 Burger and Khepera IV. The heterogeneity of the agents in *Robotic Park* along with the flexibility of its configuration are two of its main features. Most existing testbeds usually work with homogeneous agents and their systems offer very good experiences but less flexible configurations than *Robotic Park*. The architecture of *Robotic Park* allows to work in a virtual environment (simulation of the system), in a physical environment (robots in a real scenario), or in a hybrid scheme. The virtual environment includes two different simulators, based on Gazebo [28] and Webots [29]. Gazebo is the most used open-source 3D robotics simulator. It supports a wide range of sensors and objects and high compatibility with systems such as ROS and ROS2. However, it may present a certain barrier to entry for users not familiar with its environment. For this reason, *Robotic Park* includes a Webots simulator with less compatibility than Gazebo but simpler to use. It provides a friendly interface that enables programming and simulating MRS in short periods of time. This broader the user profiles who can benefit from working with our platform. Both tools allow students or researchers to develop and test algorithms before implementation with real robots.

Furthermore in the physical environment of *Robotic Park* in addition to the different types of vehicles three positioning systems that use different technologies are available: LightHouse (internal), Loco Positioning System (internal), and Vicon Motion Capture (external). All elements are connected through a ROS2 network. This system has been chosen for being more suitable when implementing decentralized architectures and allows adding new robots and sensors with minimal changes to the existing infrastructure. *Robotic Park* can also be used as a hybrid platform, i.e., to combine real and digital twin robots in an experience. This combination of agents is possible thanks to the implementation in ROS2. In fact, the implementations and experiment designs are identical for simulation and experimentation on hardware. Using the real or virtual environment only depends on the nodes or parameters that are executed, whereas the nomenclature and typology of the variables are the same in both environments. Therefore, in this work, a complete heterogeneous flexible, and easy-to-use indoor platform to perform MRS experiments is developed. The main contributions are summed up to:

- It enables the use of highly heterogeneous agents, allowing experiences with a variety of robots of different nature.
- It supports virtual, real, or hybrid scheme experiences. This expands the validation of MRS algorithms usually done by simulation to real or hybrid environments.
- It includes two hybrid frameworks that enable combining real and virtual agents in mixed reality experiences, as the digital twins are indistinguishable from the real agents. This fact allows MRS mixed reality experiences regardless of the number of agents and their nature.
- All components are integrated through ROS 2 which allows centralized, distributed, or hybrid control and communication architectures for multi-agent systems. This is a novelty regarding the widespread use of ROS in other environments and allows running distributed systems in a simpler way.
- It supports control strategies design and validation of MRS algorithms. Particularly interesting is the use of the platform for experimentation tasks based on distributive and cooperative work among agents involving restrictions in the computation time and data transmission, where event-based control techniques can be applied. In this regard, the communication between nodes is

implemented in such a way that changing from one policy to another is simple and transparent to the user. A comparison between periodic and event-based communication is presented in this paper applied to the formation control problem.
- Technical details in the development of Robotic Park, which are usually omitted in most of the publications, are also presented in the paper.

The design of *Robotic Park* was motivated by our interest in experimentation tasks with MRS involving restrictions in the computation time and data transmission. The platform has been working for one year and some experiences have already been carried out, showing its potential. Authors have studied event-based control techniques to reduce the occupancy of the communication channel [30]. It has been shown that the use of these strategies can optimize the communication and the use of the microcontrollers of the robots [31]. This is essential when working with power-constrained agents. The reduction of unnecessary operations allows the extension of the life of the batteries and extends the length of the experiences. In [32] an experience making use of mixed reality is presented. In that case, real aerial robots and mobile differential robots digital twins were combined for the formation by the consensus problem.

The paper is organized as follows. Section 2 describes in detail the experimental environment and its components. Section 3 presents the control problem of multi-agent systems focused on formation with distance constraints. Section 4 includes experiences and discusses experimental results using the platform. Finally, Section 5 ends the paper with conclusions and suggestions for future works.

II. EXPERIMENTAL PLATFORM

The platform called *Robotic Park*, as shown in Figure 1, is a testbed focused on carrying out control and robotics experiences. Its setting and components are focused on indoor applications. The target users of this platform are researchers and students. For researchers, a variety of robots with different configurations are available to validate theoretical developments in real environments. For students, this type of platform is a bonus incentive to reinforce knowledge and verify that theoretical concepts have applications in the real world.

Figure 2 shows a schematic laboratory diagram of the platform including the different components and their interactions. The platform logo is shown in the upper right corner. Components directly connected to the Robot Operating System 2 (ROS2) laboratory network are shown within the marked green area. This includes the robots (Kheperas IV, DJI Tello and Turtlebot3 Burger), a laptop to address communication with the Crazyflies and user interaction, and a personal computer (PC) to run the software Tracker 3.9 related to the Vicon MoCap system. Vicon's positioning system data is read from a node running on the laptop. As the other two positioning systems available, the Lighthouse and the Loco, are outside the ROS2 network. Their data are published through

FIGURE 1. ***Robotic Park*** **experimental platform.**

the Crazyflies which uses the Crazyradio PA connected to the laptop. Using the appropriate deck, Crazyflies can use any of the positioning systems. The Kheperas IV and DJI Tello are connected to the ROS2 network through a client-server node that can be run on the laptop or on a Raspberry Pi as a base station. The Turtlebot3 Burger uses its own Raspberry Pi to connect to the ROS2 network. All these components are described in more detail below.

A. ROS2

The ROS2 is an open-source and modular tool that helps to build robot applications. A network defined in this middleware is composed of nodes, topics, and services. Nodes are the processing units of the network with specific purposes, such as reading sensors, controlling motors, etc. Communication between nodes is carried out through topics and services. Topics are variables based on the publisher-subscriber model. When call-and-response-based communication is required, services are used.

We have chosen this framework for the following reasons: i) Its use is widespread among roboticists which increases *Robotic Park* accessibility; ii) It enables to integrate all components (software components, drivers for robots hardware, and robotic algorithms) in the same network; iii) ROS2 is multi-platform. It has first-class support for Windows, Linux, and Mac operating systems allowing seamless development of on-robot autonomy that can be programmed in C++ and Python. This includes most of the target community; iv) it allows running distributed systems in a simpler way. The last reason is the main advantage of ROS2 over its initial version, ROS, which has a centralized system architecture where all the elements processes are connected to the ROS_MASTER. In ROS2 this is no longer required. Moreover ROS Noetic, the current distribution of the original ROS, has been under development for over half a decade and will be the last. ROS2 is also compatible with more complex technologies such as the Fiware platform [33], an open-source platform for developing Internet of Things (IoT) applications.

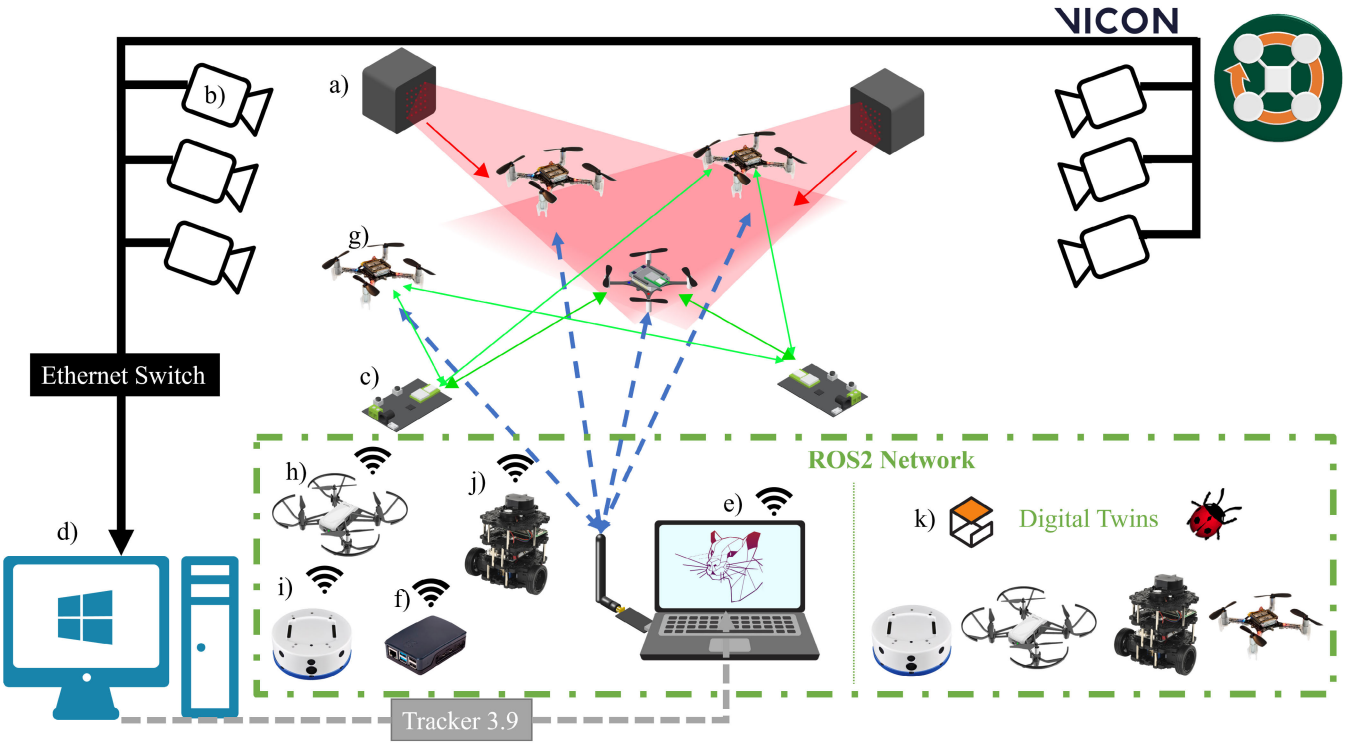

FIGURE 2. Schematic diagram of *Robotic Park* components: a) Lighthouse base stations b) Cameras Vero v2.2 c) Loco Positioning System anchors d) PC with software Tracker 3.9 e) Laptop f) Raspberry Pi as base station g) Crazuflie 2.x h) DJI Tello i) Khepera IV j) Turtlebot3 Burger k) Digital Twins of real robots.

In the designed architecture, each robot is defined within a namespace (virtual subspace of the network where all nodes and topics are prefixed in the name), and all the nodes necessary for its operation are grouped. For instance, Figure 3 shows the generic namespace of a Khepera IV robot, *agent03*. In this way, the robot set can be scaled without variable name conflicts. A driver node is defined for each robot which is responsible for the communication with sensors and actuators. If off-board controllers are available in a robot, they will be under the same namespace. Digital Twins in a ROS network are indistinguishable from real robots since they use the same nomenclature and types of nodes and topics. In software tools such as Gazebo, their appearance and sensing capabilities are configured in.urdf files [34].

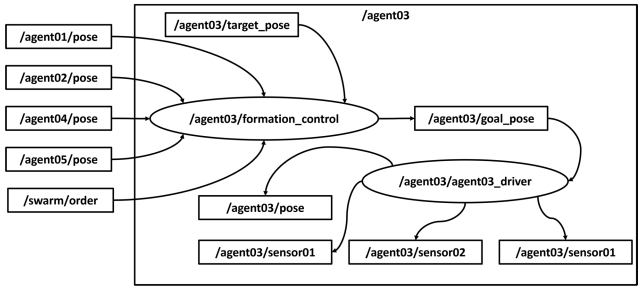

FIGURE 3. Robot namespace example.

B. POSITIONING SYSTEMS

Indoor positioning systems are indispensable to perform control experiments with any mobile robot system. One of the first issues to be considered in the design is how to obtain the location of the robots accurately. As previously mentioned, the arena (which is the volume for experimentation) is an indoor working environment. Its size is 2 m × 2 m × 2.5 m (length, width, height). To cover the positioning of robots we have three positioning systems: Vicon MoCap, LightHouse and Loco Positioning System.

The **Vicon's Motion Capture** system is an external positioning system. It estimates the position of the robots from the detection of reflective markers with a previously defined geometry. It is composed of six Vero v2.2 infrared cameras with a resolution of 2.2 megapixels. They are located around the arena in the upper area of the laboratory walls with an angle that maximizes the covered space. These cameras are connected to a PC running on the Windows operating system that executes Vicon's Tracker 3.9, the software related to the Vicon MoCap system. The software computes the position of the robot by image processing detecting a marker distribution previously predefined for each robot. The position data of each robot is published into the ROS2 network through a topic of the type *geometry_msg/Pose* using a node that incorporates the Vicon's Software Development Kit (SDK). Its precision and accuracy are below the millimeter. Its run frequency is 100 Hz (frequency of the robot position controllers), being possible to increase it up to 250 Hz.

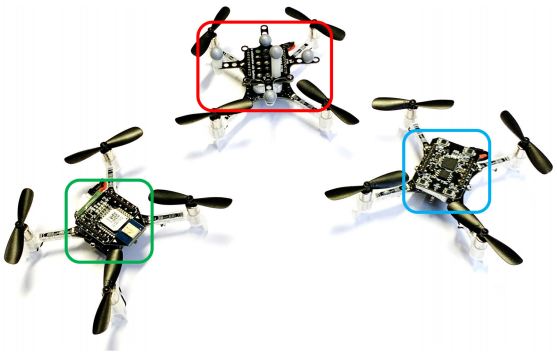

FIGURE 4. Crazyflies with Loco Positioning deck (left, green), Motion capture marker deck (center, red), and Lighthouse positioning deck (right, blue).

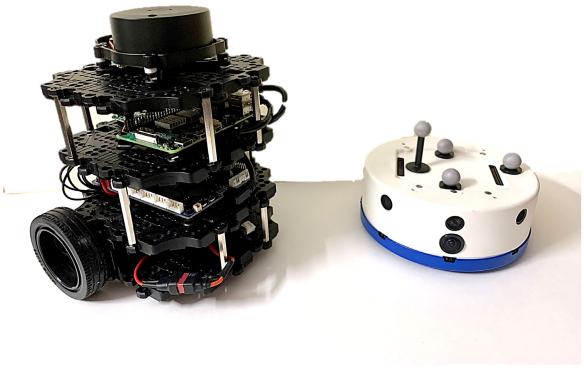

FIGURE 5. Robot Turtlebot3 Burger (left) and Khepera IV (right).

The **Lighthouse** is an internal and optically-based positioning system of the Crazyflie quadrotors. Its main advantages are the direct acquisition of the global positioning of the robot without delays due to data processing and transmission, and its low cost. The system uses the SteamVR Base Station as an optical beacon and enables an accuracy better than a decimeter and a millimeter of precision. Base stations are made up of two rotating drums (V1) or one with two inclined light planes (V2) that use laser light. The robot must be equipped with a light receiver (a photodiode) that measures the angle between the sensor and the base station. By using multiple receivers (lighthouse deck), it is possible to estimate the position and orientation of the object relative to the base station. From the known position of the base station, the position of the robot can be calculated. The system allows an operating frequency of 50 Hz with a range of 6 meters. The data is sent by the robot to the ROS2 network through a topic of the type *geometry_msg/Pose* through the Crazyradio PA.

The **Loco Positioning System** uses a similar philosophy to the LightHouse but it is based on a different technology: Ultra-Wide Band (UWB) technology instead of laser. The system supports a configuration with 4, 6, and 8 anchors strategically placed in the working environment. By sending short, high-frequency radio messages between the anchors and the robot receivers, the system measures the distance from each anchor to the receivers and calculates the receiver position from that information. It can be configured for a range of up to 30 meters with accuracy like the LightHouse system. The data is sent by the robot to the ROS2 network through a topic of the type *geometry_msg/Pose* through the Crazyradio PA.

Figure 4 shows the different decks that can be connected to the Crazyflies to use the different positioning systems described above. In the case of differential robots, due to the precision of their motors, they can use their odometry to define an initial position and the MoCap system as the external positioning system. Figure 5 shows the Turtlebot3 Burger robot using odometry and the Khepera IV robot with MoCap markers.

C. ROBOTS

The main feature of *Robotic Park* is to offer experiences for heterogeneous MRS. In its implementation, two types of the most used robots are included: aerial and differential drive robots. The description of the available robots is given below.

1) AERIAL ROBOTS

a: Crazyflie 2.x

These are classified as micro aerial vehicles. Due to their small size (92 · 92 · 29 millimeters), low mass (27 grams), and inertia, they are suitable for indoor experimentation. Crazyflie is an open-source platform with a wide range of sensors. It includes an API in Python to interact with the robot through the Crazyradio PA antenna. A maximum of 15 robots can be connected to each antenna. It has a STM32F405 microcontroller and a Bluetooth module to be controlled from a PC or a mobile device. Among its most outstanding sensors, they have a multi-ranger deck, which allows obtaining the distance to obstacles in the main axes of the robot; a Flow deck v2, for the estimation of movement in the X-Y plane by visual odometry; and an AI deck, which supposes an increase in the computing and communication through the GAP8 low-power processor.

b: DJI Tello

It is a commercial quadrotor whose main advantage over the Crazyflie is the improvement in the quality of its front camera. If we compare its physical characteristics, its size is similar (98·92.5·41 millimeters) but DJI Tello is heavier than Crazyflie (80 grams). It is a closed platform that includes a SDK in Python to interact with it. Its local positioning is done through visual odometry and a distance sensor located in its lower area. The velocity controller runs on board. To close the position control loop is necessary an external positioning system (such as Vicon Motion Capture). They use 2.4 GHz 802.11n Wi-Fi to communicate, and the robot can be configured as an Access Point to increase the number of agents.

The model of both quadrotors is the same and it only changes the physical parameters [35]. As it is a non-linear model, it is possible to carry out a large number of experiments with different types of controllers. The proposed

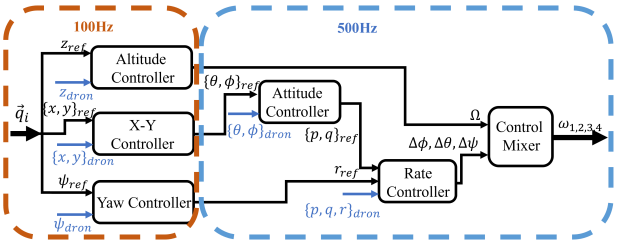

FIGURE 6. Block diagram of the quadrotors' control architecture.

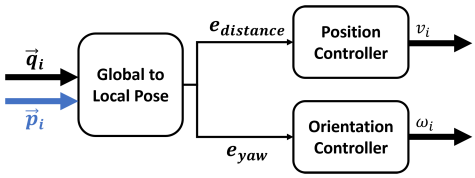

FIGURE 7. Block diagram of the differential mobile robots' control architecture.

control solutions go from advanced nonlinear algorithms [36] to simple linear PID controllers [37]. Linear controllers have better performance when the robot works around an operating point. The control architecture of this type of agents usually has two levels made up of a cascade system each one, as shown in Figure 6. The upper level is in charge of controlling the position and speed of the robot. This level usually runs at 100 Hz. Its input signal is the target position and its outputs are the thrust and the Pitch and Roll angles. The next level is the stability control and it is defined by the Attitude controller and Rate controller. It requires a higher operating frequency, 500 Hz, and its output signals are the commands that each robot's rotors must receive.

2) DIFFERENTIAL MOBILE ROBOTS

a: Khepera IV

It is a wheeled mobile robot designed by *K-Team* [38]. Its weight is 540 g and its payload limit is 2 Kg. It is specially designed to work on hard and flat surfaces indoors. It uses the odometry of its wheels to estimate its local position and its velocity. Therefore, it needs an external positioning system to know its initial position and correct the estimate if the wheels slides. Among the sensors that the robot incorporates, are a monocular color camera ($752 \cdot 480$ pixels, 30 *FPS*), 8 Infra-red proximity and ambient light sensors with up to 25 cm range, 4 Infra-red ground proximity sensors for line following applications and fall avoidance, 5 Ultrasonic sensors with range 25 cm to 2 m and 3 axis gyroscope and accelerometer are included. Communication is possible using 802.11 b/g WiFi and Bluetooth 2.0 EDR. Due to the Linux version of the robots' CPU (Yocto 1.8), communication to the ROS2 network is via WiFi with a client at a base station within the network.

b: Turtlebot3 Burger

It is an open-source wheeled mobile robot developed by ROBOTIS' Co [39]. It is one of the most used platforms for teaching robotics. It is a robot specifically designed to operate in indoor environments with ROS/ROS2. Its main advantage over the Khepera IV is its greater configuration flexibility (its main CPU is a Raspberry Pi 4 with 2Gb of RAM). Its dimensions are $138 \cdot 178 \cdot 192$ millimeters. Its weight including sensors is $1kg$ and its maximum payload is $15kg$. Its sensors in the basic configuration are a 360 Laser Distance Sensor LDS-01 or LDS-02 and an IMU with 3 axis gyroscope and accelerometer.

The full model of these robots is made up of three levels: direct kinematics, dynamic, and motors. However, due to the characteristics of these robots, we generally work with their direct kinematic model [40]. These robots are non-holonomic and subject to the following constraint:

$$\dot{x} \sin \varphi - \dot{y} \cos \varphi = 0 \tag{1}$$

where $\dot{x}$ and $\dot{y}$ denotes the linear velocities and φ the orientation of robot.

The position control architecture is made up of a single level with two controllers: position and orientation, as shown in Figure 7. The feedback signal of this control level is the position of the robot, $\vec{p}_i$, and its control signal is the linear, v_i, and angular, ω_i, velocity commands. It is necessary a first function to transform the pose value from global to local frame. The outputs of this function are the distance error, $e_{distance}$, and the yaw angle error, e_{yaw}. Obstacle avoidance is also implemented at this level. All robots on the platform have obstacle avoidance implemented by artificial potentials [41].

D. SIMULATORS

Simulators are a key tool in any experimental platform as they provide a fully controlled virtual testing and validation environment. They allow users simultaneous experimentation without occupying resources or having to schedule reservations in the real platform. Likewise, they allow experiences to be carried out when the set of users is geographically distributed.

The construction of a complete experimental platform usually requires the development of a simulator as a first step. Deploying a simulator does not require a large budget and solutions are running on different operating systems. In fact, the only constraint factor is the computational resources of the device on which they run.

Among the available tools for dynamics simulation, *Robotic Park* currently uses two of them, which, as explained below, perfectly fit our necessities: Gazebo and Webots.

1) GAZEBO

This tool is one of the most used in robotics applications because of its high integration with ROS/ROS2 [28]. Digital twins of the Khepera IV robots, DJI Tello and Turtlebot3 Burger operating with ROS2 are currently available. The Crazyflie rotors plugin is only available for ROS Noetic. Figure 8 shows a high-fidelity replica of the real laboratory

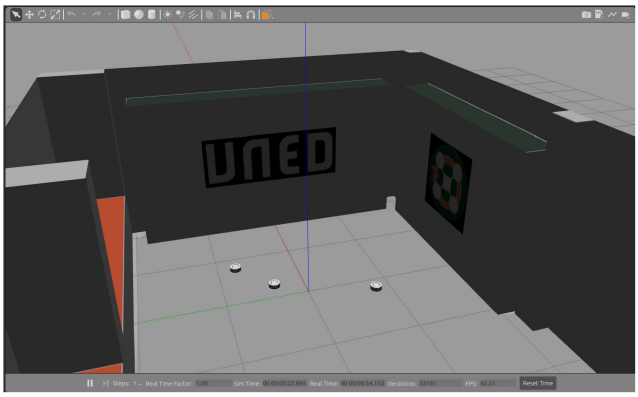

FIGURE 8. *Robotic Park* in Gazebo 11.

FIGURE 9. *Robotic Park* in Webots.

visualized using Gazebo. In this virtual environment, robots will be subject to the same spatial constraints as in the real one, and the ultrasound and laser measurements sensors will be the same in both environments. In the experiences carried out in ROS2, the simulated/real-time ratio is very close to one, which ensures that the behavior in the simulation accurately replicates the real robots.

2) WEBOTS

Webots launched in 2004 [29], provides a complete mobile robotics simulation development environment. Its use has been growing in recent years. This open-source tool was not designed to work exclusively with ROS, offering much more versatility to perform experiences outside of the ROS/ROS2 networks. In addition, it includes libraries with a large number of robot models equipped with many sensors and actuators developed by the same manufacturers. This makes researchers focus on control tasks and, hence, avoid spending a lot of time in the development of the model. Specifically, in the 2023a version, Khepera IV, Turtlebot3 Burger, and Crazyflie 2.1 robot models are available.

III. MULTIAGENT CONTROL ARCHITECTURE

MRS robots are provided with an advanced control system usually comprised of several hierarchically ordered subsystems. In this paper, a three-level control structure of each agent is used as illustrated in Figure 10: individual controller at the low level, coordination at the middle level, and path planning at the upper level. The first level (individual controller) takes care of the position and orientation of each agent, and its control architecture has already been presented in section II-C. Additionally, the control law includes at this level a term to avoid collisions between robots. Its implementation is based on repulsive potential fields as follows:

$$U_k = \begin{cases} \frac{1}{2}\eta(\frac{1}{d_k} - \frac{1}{d_0})^2 & if\ d_k \leq d_0 \\ 0 & if\ d_k > d_0 \end{cases} \tag{2}$$

where U_k denotes the repulsive potential of sensor k, d_k is the value of the distance between the robot and the obstacle, d_0 is a threshold that activates the repulsive potential, and η is a constant that characterizes the field. Then, the resulting repulsive force F_k can be written as follows:

$$F_k = -\nabla U_k = \begin{cases} \eta(\frac{1}{d_k} - \frac{1}{d_0})\frac{1}{d_k^2}\frac{p_k - p_o}{d_k} & if\ d_k \leq d_0 \\ 0 & if\ d_k > d_0 \end{cases} \tag{3}$$

where $p_k - p_o$ is the relative position between the robot and the obstacle. Then, the sum of all repulsive forces is $F = \sum_k F_k$, and hence, this has an impact on the goal position according to the following expression:

$$u^{oa} = h \cdot v \cdot \frac{F}{\|F\|} \tag{4}$$

where u^{oa} is the deviation of the goal position signal received from the coordination level, h is the period of the controller, and v is a constant velocity.

The coordination control of the agents corresponds to the second level and can be implemented either with a centralized or decentralized control architecture [42]. In centralized control, the global controllers are implemented on a single node executed on a PC. This node is subscribed to all robot positions and generates the goal positions for each one. This configuration improves the performance regarding agent synchronization. However, this architecture presents serious scalability problems when the number of agents increases. In the decentralized control scheme, each agent has its own onboard controller in its namespace, and it is only subscribed to the position of a subset of the agents called *neighbors*. This architecture increases the practical autonomy of the agents and allows the onboard implementation of the controller, and it further reduces delays when transmitting the goal position to the position controllers.

The controllers that are involved at the Coordination level are the formation controller, in charge of generating the commands so that the desired formation is achieved, and the tracking control [43], which will command the formation in a coordinated way over the space.

Regardless of whether the architecture is centralized or distributed, the multi-agent system can be modeled in terms of a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_N\}$ is a finite set of N vertices representing the nodes or agents and $\mathcal{E} \subseteq V \times V$ is a finite set of edges, representing the communication links

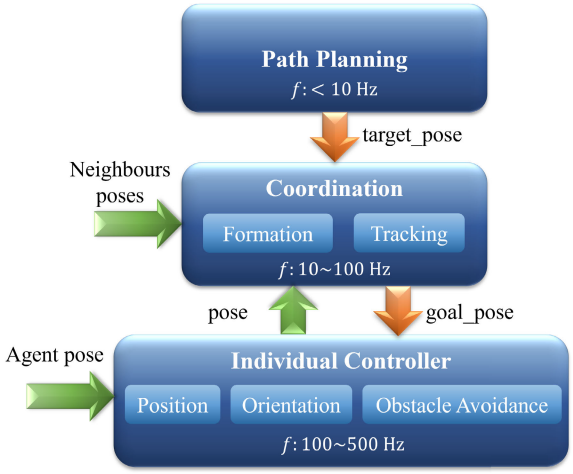

FIGURE 10. Hierarchical structure of MRS' agent control.

between a pair o agents. An edge connecting nodes $i \in V$ and $j \in V$ will be denoted as (i, j). Any two nodes (agents) i and j are adjacent if they are connected by an edge (i, j) and $\mathcal{E} = \{(i, j) \in \mathcal{V} \times \mathcal{V} : v_i, v_j \; adjacent\}$. If no edge has orientation, that is, if all edges (communication links) in the graph are bidirectional, the graph is undirected, $(i, j) \in \mathcal{E} \iff (j, i) \in \mathcal{E}$. On the other hand, when any of the connections is not bidirectional, the graph is directed. The *adjacency matrix*, $A = [a_{ij}] \in R^{N \times N}$, of $\mathcal{G}$ is defined by $a_{ij} = 1$ if i and j are adjacent or $a_{ij} = 0$ otherwise. Furthermore, the *degree matrix* $D = d_{ij} \in R^{N \times N}$, of $\mathcal{G}$ is the diagonal matrix defined as $d_{ij} = deg(v_i)$ if $i = j$ or $d_{ij} = 0$ otherwise, where the degree $deg(v_i)$ of a vertex counts the number of nodes connected to the vertex i. In other words, $deg(v_i)$ is the cardinality of node i*'s neighbor set* $N_i = \{j \in \mathcal{V} : (i, j) \in \mathcal{E}\}$. The *Laplacian matrix* L of $\mathcal{G}$ is defined as follows:

$$L = D - A. \tag{5}$$

There are several ways in the literature to approach the multi-agent formation problem, as surveyed in [44]. Moreover, the formation can be basically defined in two ways: a desired relative position for each pair of agents that are connected in the graph, so that each edge will have a target relative position defined by a vector; or a target distance d_{ij} to be reached for each pair of connected agents i and j. This second approach fits better with the sensing capabilities that autonomous robots usually have, but it should be handled with more care since further properties such as rigidity of the graph should be considered [45].

Let us denote as $p_i \in \mathbb{R}^3$ the position of any agent i. The formation controller is defined as follows:

$$u_i^s = \frac{1}{2} \sum_{j \in N_i} \mu_{ij} (d_{ij}^2 - \|p_i - p_j\|^2)(p_i - p_j) \tag{6}$$

where $\mu_{ij} > 0$ is a gain and d_{ij} represents the desired distance between agents i and j. Note that the control signal u_i^s approaches zero when the distance between i and j given by $\|p_i - p_j\|$ converges to d_{ij}.

From the practical point of view, defining the formation in terms of desired distances and not in terms of desired relative positions makes the movement in the three axes to be coupled. This implies that a disturbance in any of the axes causes the controller to generate a new control signal for each one.

At the coordination level, the tracking controller receives the topic *target_pose*, denoted by q_i, as a reference signal. This input comes from the immediately higher level responsible for the agent's path planning. Then, the generated control signal generated by the tracking controller, u_i^t, is computed in the following form:

$$u_i^t = -\frac{1}{2} k_i \|p_i - q_i\|^2 (p_i - q_i) \tag{7}$$

where $k_i > 0$ is a gain. This control signal is added to u_i^s defined above in (6), obtaining a new goal position, and yields the control input for the agent i as follows:

$$u_i = u_i^s + u_i^t. \tag{8}$$

Moreover, there are several ways to generate this target position q_i in the coordinated movement of a multi-agent system. One way is to define a leader of the formation that will follow a trajectory generated by the upper level of the control architecture. In this leader-follower scheme, the leader transmits its position to its neighbors at certain instants of time, but it does not receive information from them. This way, the team of robots tries to follow the leader, and the agents that are further away from it in the communication graph will have a slower response. This implementation shows a directed graph. Another strategy is called dominance formation [43], in which a weight is defined for each link, and those edges that connect to the leader have a higher weight. In this case, the leader receives its neighbors' position. Finally, if there is no leader agent in the formation, the collective movement is achieved from the consensus between all the agents.

A. EVENT BASED CONTROL

Event-based sampling and control is an alternative to the periodic sampling method that has been proven to be effective in reducing the number of samples, updating the control signal, etc [46]. The main idea is that it is the state/the output of the system and not the time what determines when to sample the system. The average rate of event-triggered transmissions is usually much lower than the sampling frequency in sampled-data systems, which implies the superiority of the event-based techniques in resource-constrained applications. However, the sampled-data theory is no longer applicable for the event-triggered control since the fundamental assumption of equidistant sampling is violated, and new theoretical developments have been necessary to derive stability conditions (see [47] and references therein).

In recent decades it has experienced a boom due to the advantages it provides in cyber-physical systems (networked control systems). In these systems, there is a strong coupling between control, computation, and communication. Therefore, the design of strategies must be carried out in an

integrated manner to obtain adequate results. In the case of battery-powered systems, such as robots, the autonomy of the agents is a bottleneck in their development. Efficient use of available resources has a particular impact as it allows the extension of the system's autonomy.

Normally, in event-based sampling, a trigger function is defined depending on an error function and a threshold, so that an event (sampling) occurs when the error function reaches that limit value. Usually, the error function is defined as an absolute difference between the last transmitted measurement and the current measurement, and there are different proposals for the threshold in the literature (constant, state-dependent, etc) but, in general terms, the larger the threshold, the lower the rate of events. Thus, the trigger time of an event-based control system is defined by the following recursive form:

$$t_{k+1} = inf\{t : t > t_k, f(e(t), x(t)) > 0\} \tag{9}$$

where $x(t)$ denotes the state of the system at time t, $e(t) = x(t_k) - x(t)$ the error with respect to the state at the last sampling instant (t_k) and $f(e(t), x(t))$ is the trigger function.

In the case of coordination control in multi-agent systems, the event-based mechanism is implemented to reduce the communication between agents, so that an agent transmits an updated measurement (its position) to its neighbors when the trigger function $f(e(t), x(t))$ crosses zero. In other words, this trigger condition sets a maximum threshold for the error so that movements of the agent below this threshold will not be informed to the neighbors. When this implementation is used to decrease the number of controller updates and, therefore, to reduce the computational load, the trigger condition is linked to the error received by the controller.

IV. EXPERIMENT AND RESULTS

In this section, we describe some of the first tests conducted to show the potential uses of the developed platform. It is a mixed reality experience that involves using digital twins. The MRS is composed of different robots: 3 real Crazyflie 2.1, 1 real Turtlebot3 Burger, and 3 Khepera IV digital twins. We will address a problem of formation control where the agents are bound together maintaining a rigid formation while fixing constant the hover height of the Crazyflies. We show the fundamental parameters of the MRS used.

A. COMMUNICATION GRAPH

The directed and connected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ that defines the communication between the seven agents is shown in Figure 11. The set of nodes:

$$\mathcal{V} = \{TB01, KH01, KH02, KH03, CF01, CF02, CF03\} \tag{10}$$

represents the robots, where the prefix "TB" refers to the Turtlebot3 Burger, "KH" to the Khepera IV and "CF" to the Crazyflie 2.x.

The Turtlebot3 Burger is considered a "leader" and the communication with its neighbors is directed. The links between the agents are defined by the following adjacency matrix:

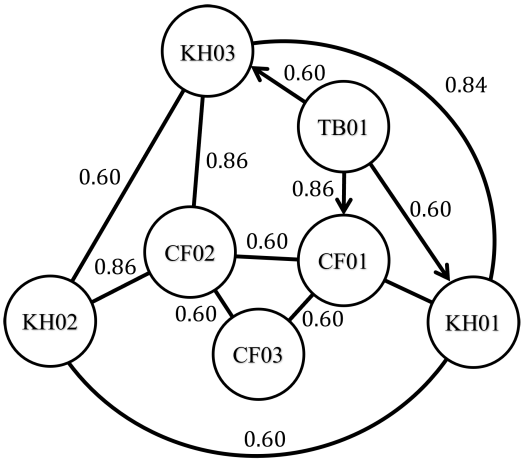

FIGURE 11. MRS Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$.

$$A = \begin{bmatrix} 0&0&0&0&0&0&0 \\ 1&0&1&1&1&0&0 \\ 0&1&0&1&0&1&0 \\ 1&1&1&0&0&1&0 \\ 1&1&0&0&0&1&1 \\ 0&0&1&1&1&0&1 \\ 0&0&0&0&1&1&0 \end{bmatrix}. \tag{11}$$

The desired formation is defined in terms of target distances d_{ij}. The values of d_{ij} for each edge are depicted over the communication graph in Figure 11.

B. EXPERIMENTS

In the formation control experiment carried out in this work it is intended that all agents in the formation follow the trajectory described by the TB01 while maintaining the formation specified by the set of desired distances for each edge of the graph as shown in Figure 11.

The goal of this experience is to analyze the convergence speed of the formation and the effect of the leader's motion over the formation. An event-based sampling communication strategy (see section III-A for details) will be compared with a traditional periodic sampling. The trigger function used when working with the event-based scheme is send-on-delta based on its current position with 1 cm as constant threshold value [48].

Two different cases are detailed:

- **Case 1**. Each agent communicates its position to its neighbors at 50 Hz.
- **Case 2**. Each agent communicates its position to its neighbors when the event-based trigger is activated.

So, in the first case, the control law (6) will be updated periodically according to the new neighbors' poses received. In the second case, the controller will be updated whenever an event occurs in the neighborhood or at the agent itself. In both cases, the experiment consists of two steps: First, the agents

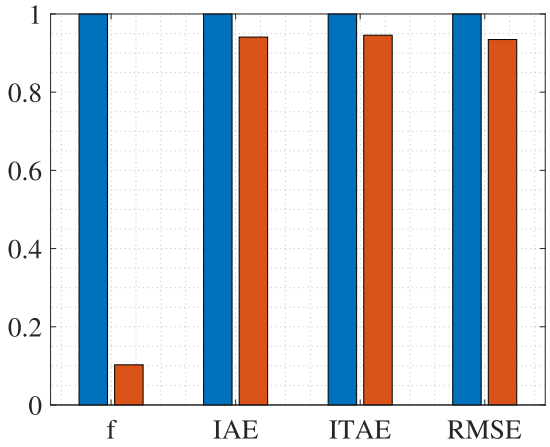

FIGURE 12. Global results. Case 1 (blue), Case 2 (red).

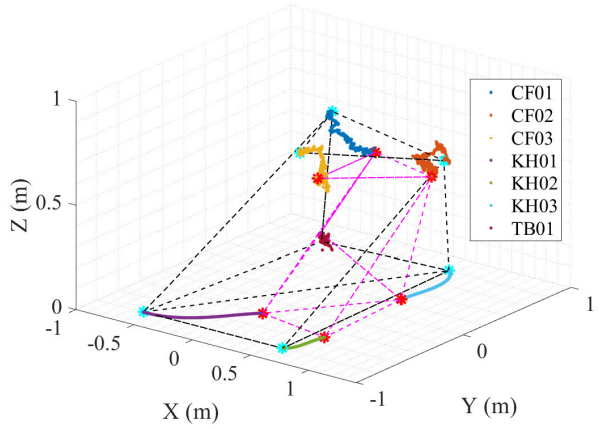

FIGURE 13. Formation progress before leader displacement from initial poses (black) to desire constraints formation (magenta). Initial position: *; Final position: *.

start moving to acquire the desired formation (this is achieved approximately in 12 s); and second, the agent leader moves autonomously 0.75 m (on the x axis following a straight line). The initial position and orientation of each agent could be random. However, in the two cases described below, the initial positions are the same to compare the results. The experiment starts when the drones have taken off and reached a stable position around 0.8 m.

To evaluate quantitatively the performance of the two strategies, we use the following performance indices:

- **Average frequency (f)**. With this parameter, we analyze the number of transmissions carried out by the agents over the time of the experiment. This index shows the flow of information through the communication channel.
- **Integral Absolute Error (IAE)**. This index weights all errors equally over time. It gives global information about the agents.
- **Integral of Time-weighted Absolute Error (ITAE)**. In systems that use step inputs, the initial error is always high. Consequently, to make comparisons between systems, it is more relevant than the errors that are maintained over time have a greater weight than the initial errors.
- **Root Mean Square Error (RMSE)**. This performance index is used to assess the accuracy of the model in terms of the error variance.

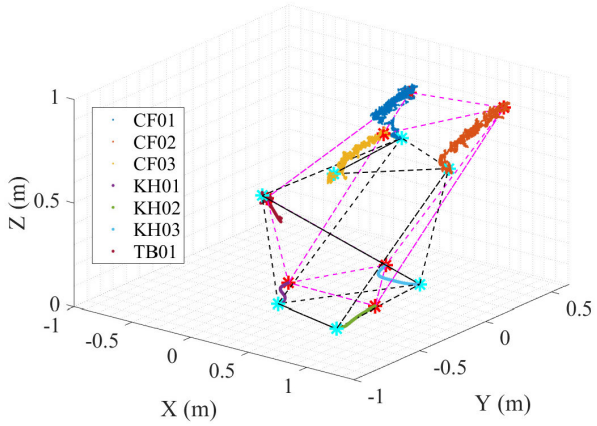

FIGURE 14. Formation progress after leader displacement from disturbances links (black) to desire constraints formation (magenta). Initial position: *; Final position: *.

C. RESULTS

The global results of the performance measures described above are shown in Figure 12. The purpose is to demonstrate the potential of the *Robotic Park* platform and the impact of event-based communication versus continuous communication. To improve the comparison, all results are normalized with respect to the reference case, case 1. Hence, all performance indices take a unit value for case 1, and the results for the second case are weighed accordingly. This provides a clearer interpretation of the results.

Upon analyzing the obtained errors, a slight improvement can be seen in the second case (around 5%). This is mainly due to the filtering effect that event-based communication performs by removing the ripple in the signal transmitted below the trigger threshold. This smoothness of the signal results in a similar smoothness of the control signals obtained in the formation control of each agent. However, a major improvement is obtained over the average frequency, as expected. The observed decrease in transmitted signals is 89% (average frequency 5.07 Hz). This significant reduction results in a more efficient operation of the MRS, whereas it does not have a negative impact on the performance. In addition, for those agents with critical energy capacity, such as the Crazyflies, reducing communication channel usage means an increase in their autonomy and the range of feasible experiences.

Figure 15 shows the results obtained for each of the agents analyzing the total error calculated in the intra-swarm controller. A similar error evolution is observed in the overall results. In the case of the sampling frequencies, we observe that the improvement is greater in the case of the ground robots (96%, average frequency 1.57 Hz) than in the case of

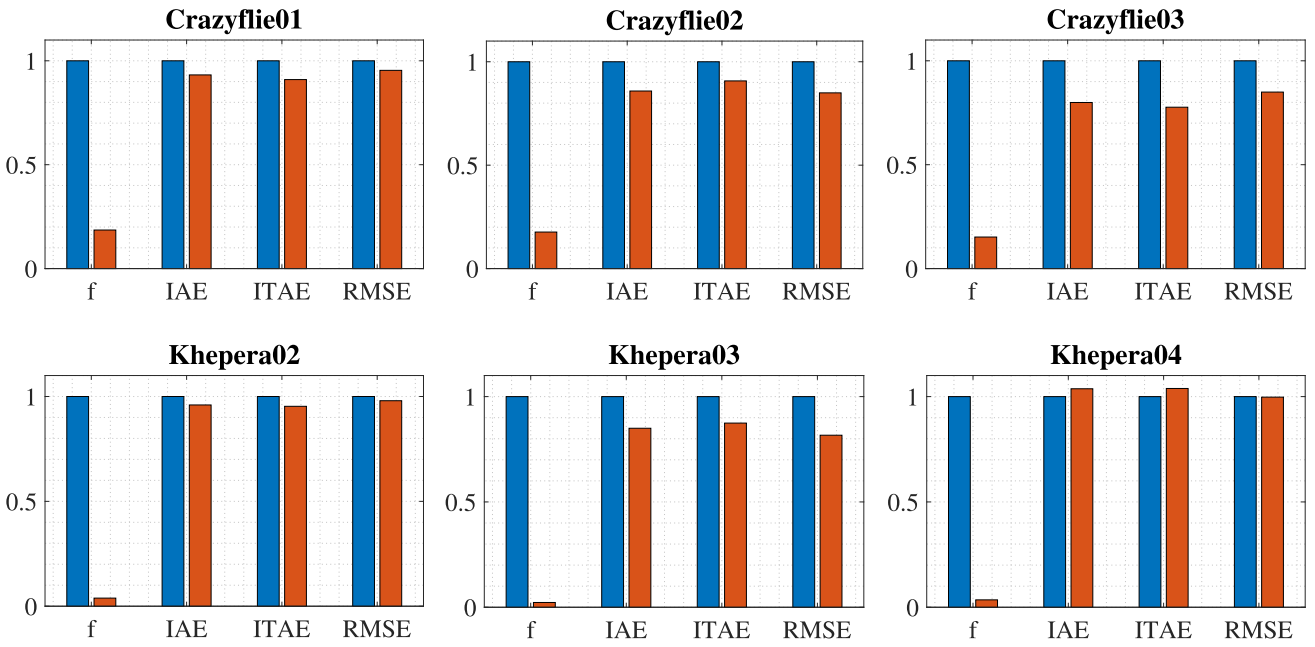

FIGURE 15. Individuals results. Case 1 (blue), Case 2 (red).

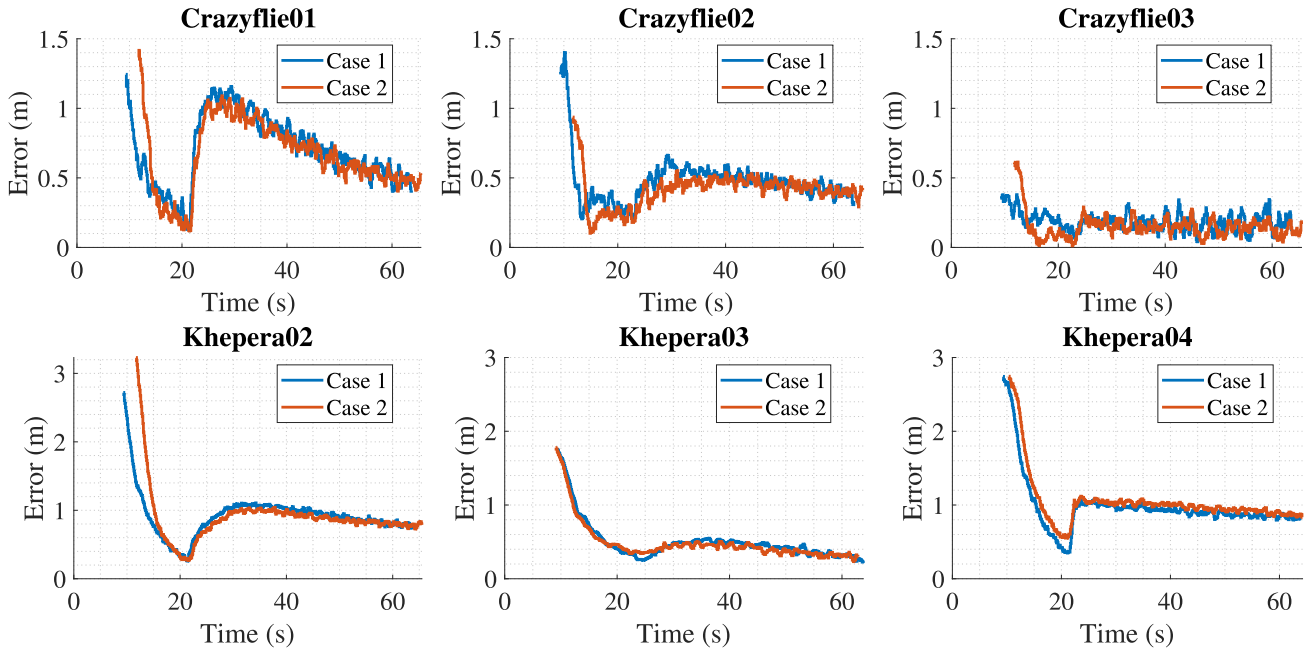

FIGURE 16. Global Errors.

the drones (81%, average frequency 8.58 Hz). This is due to the more stable and smooth dynamics of the mobile robots, which prevents event generation when they reach the desired positions.

In Figure 16, the evolution over time of the error for each agent is shown. The two steps of the experiment can be clearly visualized in the Figure: First, the movement to reach the formation (time 9s) and then the movement of the leader that makes the rest of the system move accordingly (time 21s). In both cases, a decreasing trend towards zero error is observed. In the case of the formation acquisition, all agents respond in an equivalent way. Figure 13 shows the 3D trajectories followed by the agents. However, the displacement of the leader has a greater impact on those agents directly connected to it, whereas the response of those agents that are further away in the graph is smoother and slower. This also makes the convergence to the formation slower. Figure 14 shows the 3D trajectories in this case.

V. CONCLUSION

This work presents the experimental platform *Robotic Park* for MRS research in control and robotics fields. The platform allows experiences in three modes: using virtual agents, real agents, or combining both. The hardware and software components of the platform aim to achieve a flexible and versatile testbed. To facilitate the development of experiences using *Robotic Park*, a complete description of different types of robots including their control architecture and the key features of the positioning systems available are given. Moreover, the developed simulation environments which accurately replicate real scenery have been presented. They allow an efficient experience design before implementation in the real system. An MRS experience that combines real robots and digital twins in formation and displacement tasks has been included to reflect the potential of the platform. This experience illustrates a case-of-use of the system as well as the performance of the proposed control approaches in terms of accuracy and computational efficiency.

Future works involve adding robots with different constraints such as omnidirectional wheeled, spherical mobile robots or Ackermann steering mechanisms. In the same way, a multi-camera vision-based system will be evaluated as a complement to the available positioning systems. Additionally, due to the distance learning format of the UNED, with the high geographical distribution of the students, it is planned to enable remote access to the platform. This will enrich our existing network of remote laboratories. Finally, the authors are currently preparing to release the source code of the simulators to be publicly available.

REFERENCES

[1] S. G. Tzafestas, "Mobile robot control and navigation: A global overview," *J. Intell. Robot. Syst.*, vol. 91, no. 1, pp. 35–58, 2018.

[2] S. García, M. E. López, R. Barea, L. M. Bergasa, A. Gómez, and E. J. Molinos, "Indoor SLAM for micro aerial vehicles control using monocular camera and sensor fusion," in *Proc. Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Bragança, Portugal, May 2016, pp. 205–210.

[3] S. Knorn, Z. Chen, and R. H. Middleton, "Overview: Collective control of multiagent systems," *IEEE Trans. Control Netw. Syst.*, vol. 3, no. 4, pp. 334–347, Dec. 2016.

[4] A. R. Cheraghi, S. Shahzad, and K. Graffi, "Past, present, and future of swarm robotics," in *Proc. SAI Intell. Syst. Conf.*, vol. 3, Cham, Switzerland, 2022, pp. 190–233.

[5] J. Hu, P. Bhowmick, I. Jang, F. Arvin, and A. Lanzon, "A decentralized cluster formation containment framework for multirobot systems," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1936–1955, Dec. 2021.

[6] R. Duque Estrada, F. Kannenberg, H. J. Wagner, M. Yablonina, and A. Menges, "Spatial winding: Cooperative heterogeneous multi-robot system for fibrous structures," *Construct. Robot.*, vol. 4, nos. 3–4, pp. 205–215, Dec. 2020.

[7] J. Stephan, J. Fink, V. Kumar, and A. Ribeiro, "Concurrent control of mobility and communication in multirobot systems," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1248–1254, Oct. 2017.

[8] S. Tomer, C. Kitts, M. Neumann, R. McDonald, S. Bertram, R. Cooper, M. Demeter, E. Guthrie, E. Head, A. Kashikar, J. Kitts, M. London, A. Mahacek, R. Rasay, D. Rajabhandharaks, and T.-Y. Yeh, "A low-cost indoor testbed for multirobot adaptive navigation research," in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, USA, Mar. 2018, pp. 1–12.

[9] S. Guler, M. Abdelkader, and J. S. Shamma, "Infrastructure-free multi-robot localization with ultrawideband sensors," in *Proc. Amer. Control Conf. (ACC)*, Philadelphia, PA, USA, Jul. 2019, pp. 13–18.

[10] F. Li, S. Yang, X. Yi, and X. Yang, "CORB-SLAM: A collaborative visual slam system for multiple robots," in *Proc. Int. Conf. Collaborative Comput., Netw., Appl. Worksharing*, Edinburgh, U.K., 2017, pp. 480–490.

[11] P. Anggraeni, A. S. Sunarya, and Z. N. Sani, "Formation tracking control simulation of multi robot systems for omni-wheels robot," in *Proc. 3rd Int. Symp. Mater. Electr. Eng. Conf. (ISMEE)*, Bandung, Indonesia, Nov. 2021, pp. 160–165.

[12] J. Hu and A. Lanzon, "Cooperative adaptive time-varying formation tracking for multi-agent systems with LQR performance index and switching directed topologies," in *Proc. IEEE Conf. Decis. Control (CDC)*, Miami, FL, USA, Dec. 2018, pp. 5102–5107.

[13] J. A. V. Trejo, M. A. Medina, J. A. V. Trejo, C. D. G. Beltrán, and J. G. Morales, "Robust observer-based leader-following consensus for multi-agent systems," *IEEE Latin Amer. Trans.*, vol. 19, no. 11, pp. 1949–1958, Nov. 2021.

[14] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Online trajectory generation with distributed model predictive control for multi-robot motion planning," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 604–611, Apr. 2020.

[15] J. Yu, J. A. Vincent, and M. Schwager, "DiNNO: Distributed neural network optimization for multi-robot collaborative learning," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 1896–1903, Apr. 2022.

[16] M. Fan, J. He, S. Ding, Y. Ding, M. Li, and L. Jiang, "Research and implementation of multi-robot path planning based on genetic algorithm," in *Proc. 5th Int. Conf. Autom., Control Robots (ICACR)*, Nanning, China, Sep. 2021, pp. 140–144.

[17] H. Park, A. Easwaran, and S. Andalam, "TiLA: Twin-in-the-loop architecture for cyber-physical production systems," in *Proc. IEEE 37th Int. Conf. Comput. Design (ICCD)*, Abu Dhabi, United Arab Emirates, Nov. 2019, pp. 82–90.

[18] J. Collins, S. Chand, A. Vanderkop, and D. Howard, "A review of physics simulators for robotic applications," *IEEE Access*, vol. 9, pp. 51416–51431, 2021.

[19] S. Wilson, P. Glotfelter, L. Wang, S. Mayya, G. Notomista, M. Mote, and M. Egerstedt, "The robotarium: Globally impactful opportunities, challenges, and lessons learned in remote-access, distributed control of multirobot systems," *IEEE Control Syst.*, vol. 40, no. 1, pp. 26–44, Feb. 2020.

[20] L. Paull et al., "Duckietown: An open, inexpensive and flexible platform for autonomy education and research," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Singapore, May 2017, pp. 1497–1504.

[21] J. M. Allen, R. Joyce, A. G. Millard, and I. Gray, "The Pi-puck ecosystem: Hardware and software support for the e-puck and e-puck2," in *Proc. Int. Conf. Swarm Intell.* Cham, Switzerland: Springer, 2020, pp. 243–255.

[22] W. Jo, J. Kim, R. Wang, J. Pan, R. K. Senthilkumaran, and B.-C. Min, "SMARTmBOT: A ROS2-based low-cost and open-source mobile robot platform," 2022, *arXiv:2203.08903*.

[23] P. A. F. Rezeck, H. Azpurua, and L. Chaimowicz, "HeRo: An open platform for robotics research and education," in *Proc. Latin Amer. Robot. Symp. (LARS) Brazilian Symp. Robot. (SBR)*, Curitiba, Brazil, Nov. 2017, pp. 1–6.

[24] W. Hoenig, C. Milanes, L. Scaria, T. Phan, M. Bolas, and N. Ayanian, "Mixed reality for robotics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Hamburg, Germany, Sep./Oct. 2015, pp. 5382–5387.

[25] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital twin in manufacturing: A categorical literature review and classification," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1016–1022, 2018.

[26] W. Giernacki, M. Skwierczynski, W. Witwicki, P. Wronski, and P. Kozierski, "Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering," in *Proc. 22nd Int. Conf. Methods Models Autom. Robot. (MMAR)*, Miedzyzdroje, Poland, Aug. 2017, pp. 37–42.

[27] W. Giernacki, J. Rao, S. Sladic, A. Bondyra, M. Retinger, and T. Espinoza-Fraire, "DJI tello quadrotor as a platform for research and education in mobile robotics and control engineering," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Dubrovnik, Croatia, Jun. 2022, pp. 735–744.

[28] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sendai, Japan, Sep./Oct. 2004, pp. 2149–2154.
[29] O. Michel, "Cyberbotics ltd. Webots^{TM}: Professional mobile robot simulation," *Int. J. Adv. Robot. Syst.*, vol. 1, no. 5, pp. 39–42, 2004.
[30] F. J. Mañas-Álvarez, R. Dormido, M. Guinaldo, R. Socas, and S. Dormido, "A vision based navigation platform for control learning," *IFAC-PapersOnLine*, vol. 55, no. 17, pp. 138–143, 2022.
[31] F. J. Mañas-Álvarez, R. Dormido, M. Guinaldo, R. Socas, and S. Dormido, "Control de formación basado en eventos para crazyflie 2.1," in *Proc. Simposio Conjunto de los Grupos Temáticos del Comité Español de Automática: Modelado, Simulación, Optimización e Ingeniería de Control*, Burgos, Spain, 2022, pp. 47–52.
[32] F. J. Mañas-Álvarez, M. Guinaldo, R. Dormido, R. Socas, and S. Dormido, "Formation by consensus in heterogeneous robotic swarms with twins-in-the-loop," in *Proc. 5th Iberian Robot. Conf. (ROBOT)*, in Lecture Notes in Networks and Systems, vol. 589. Cham, Switzerland: Springer, 2023, pp. 435–447.
[33] S. Rautiainen, M. Pantano, K. Traganos, S. Ahmadi, J. Saenz, W. M. Mohammed, and J. L. M. Lastra, "Multimodal interface for human–robot collaboration," *Machines*, vol. 10, no. 10, pp. 957–980, 2022.
[34] K. Takaya, T. Asai, V. Kroumov, and F. Smarandache, "Simulation environment for mobile robots testing using ROS and gazebo," in *Proc. 20th Int. Conf. Syst. Theory, Control Comput. (ICSTCC)*, Sinaia, Romania, Oct. 2016, pp. 96–101.
[35] Y. Feng, C. A. Rabbath, S. Rakheja, and C.-Y. Su, "Adaptive controller design for generic quadrotor aircraft platform subject to slung load," in *Proc. IEEE 28th Can. Conf. Electr. Comput. Eng. (CCECE)*, Halifax, NS, Canada, May 2015, pp. 1135–1139.
[36] M. S. Esmail, M. H. Merzban, A. A. M. Khalaf, H. F. A. Hamed, and A. I. Hussein, "Attitude and altitude nonlinear control regulation of a quadcopter using quaternion representation," *IEEE Access*, vol. 10, pp. 5884–5894, 2022.
[37] J. Li and Y. Li, "Dynamic analysis and PID control for a quadrotor," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Beijing, China, Aug. 2011, pp. 573–578.
[38] *Khepera IV User Manual*, KTeam, Cham, Switzerland, 2015.
[39] R. Amsters and P. Slaets, "Turtlebot 3 as a robotics education platform," in *Proc. Int. Conf. Robot. Educ. (RiE)*. Cham, Switzerland: Springer, 2020, pp. 170–181.
[40] E. Fabregas, G. Farias, S. Dormido-Canto, M. Guinaldo, J. Sánchez, and S. D. Bencomo, "Platform for teaching mobile robotics," *J. Intell. Robot. Syst.*, vol. 81, no. 1, pp. 131–143, 2016.
[41] G. Li, A. Yamashita, H. Asama, and Y. Tamura, "An efficient improved artificial potential field based regression search method for robot path planning," in *Proc. IEEE Int. Conf. Mechatronics Autom.*, Chengdu, China, Aug. 2012, pp. 1227–1232.
[42] A. Gautam and S. Mohan, "A review of research in multi-robot systems," in *Proc. IEEE 7th Int. Conf. Ind. Inf. Syst. (ICIIS)*, Chennai, India, Aug. 2012, pp. 1–5.
[43] S. Kimathi and B. Lantos, "Unmanned aerial vehicles swarm flocking architectures: An overview," in *Proc. IEEE 10th Jubilee Int. Conf. Comput. Cybern. Cyber-Med. Syst. (ICCC)*, Reykjavík, Iceland, Jul. 2022, pp. 383–388.
[44] K. K. Oh, M. C. Park, and H. S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, Mar. 2015.
[45] Z. Sun, Q. Liu, N. Huang, C. Yu, and B. D. O. Anderson, "Cooperative event-based rigid formation control," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 7, pp. 4308–4320, Jul. 2021.
[46] W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *Proc. IEEE 51st IEEE Conf. Decis. Control (CDC)*, Maui, HI, USA, Dec. 2012, pp. 3270–3285.
[47] E. Aranda-Escolástico, M. Guinaldo, R. Heradio, J. Chacón, H. Vargas, J. Sánchez, and S. Dormido, "Event-based control: A bibliometric analysis of twenty years of research," *IEEE Access*, vol. 8, pp. 47188–47208, 2020.
[48] V. Vasyutynskyy and K. Kabitzsch, "A comparative study of PID control algorithms adapted to send-on-delta sampling," in *Proc. IEEE Int. Symp. Ind. Electron.*, Bari, Italy, Jul. 2010, pp. 3373–3379.

FRANCISCO-JOSÉ MAÑAS-ÁLVAREZ received the B.S. degree in industrial electronics engineering and the M.S. degree in industrial engineering from the University of Almería, Almería, Spain, in 2019, and the M.S. degree in systems and control engineering from the National Distance Education University (UNED), in 2022, where he is currently pursuing the Ph.D. degree with the Department of Computer Sciences and Automatic Control. His research interests include mobile robot control, robot navigation, multi-agent systems, and engineering education.

MARÍA GUINALDO received the B.S. degree in computer engineering and the M.S. degree in physics from the University of Salamanca, Salamanca, Spain, in 2008, and the Ph.D. degree in computer engineering from the National Distance Education University (UNED), in 2013. She is currently an Associate Professor with the Department of Computer Sciences and Automatic Control, UNED. Her research interests include networked control systems, event-based control, multi-agent systems, and engineering education.

RAQUEL DORMIDO received the degree in physics from the Complutense University of Madrid, in 1995, and the Ph.D. degree in science from the National Distance Education University (UNED), in 2001. Since 1995, she has been with the Department of Computer Sciences and Automatic Control, UNED, as an Assistant Professor. Her research interests include the modeling and simulation of industrial processes, machine learning, and the design of systems for control education.

SEBASTIÁN DORMIDO (Member, IEEE) received the B.S. degree in physics from the Complutense University of Madrid, Madrid, Spain, in 1968, the Ph.D. degree in science from Basque Country University, Bilbao, Spain, in 1971, and the Doctor Honorary degree from Universidad de Huelva and Universidad de Almería. In 1981, he was appointed as a Professor in control engineering with National Distance Education University (UNED), Madrid. He has authored or coauthored more than 250 technical papers in international journals and conferences and has supervised more than 35 Ph.D. theses. His research interests include the computer control of industrial processes, model-based predictive control, hybrid control, and web-based laboratories for distance education. From 2001 to 2006, he was the President of the Spanish Association of Automatic Control, CEA-IFAC. He received the National Automatic Control Prize from the IFAC Spanish Automatic Control Committee.