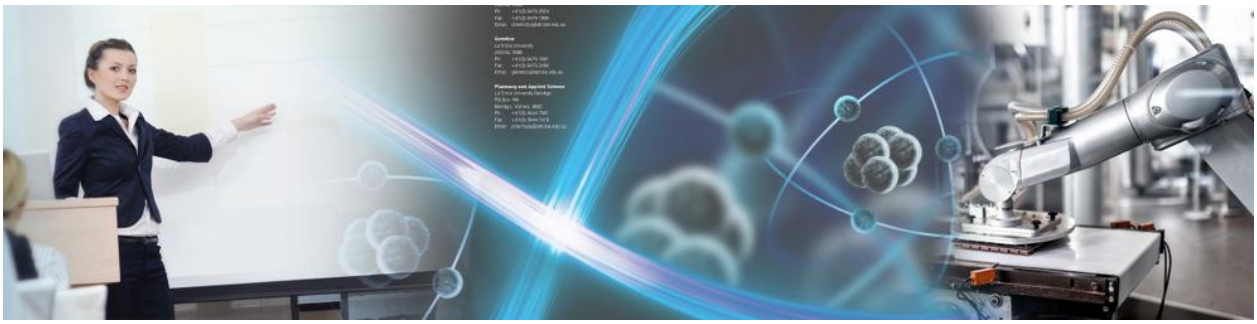
Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Кафедра комп'ютерно-інтегрованих технологій, автоматизації, робототехніки та безпекової інженерії

X Міжнародна Конференція
ВИРОБНИЦТВО
&
МЕХАТРОННІ СИСТЕМИ 2026



X International Conference
MANUFACTURING
&
MECHATRONIC SYSTEMS 2026

M&MS

2026

X International Conference

22-23 October

Kharkiv

УДК: 005:004.896:62-65:338.3

Виробництво & Мехатронні Системи 2026: матеріали X-ої Міжнародної конференції, Харків, 22-23 жовтня 2026 р.: тези доповідей / [редкол. І.Ш. Невлюдов (відповідальний редактор)].-Харків: [електронний друк], 2026. – 87 с.

У збірник включені тези доповідей, які присвячені сучасним тенденціям розвитку технологій та засобів виробництва та мехатронних систем, передовому досвіду та впровадженню їх в галузях систем промислової автоматизації та керування виробництвом; системній інженерії; CAD/CAM/CAE системах; мехатроніці (електро-механічних системах, електронних інструментах систем керування, механічних CAD системах); робототехніці та засобахм інтелектуалізації; MEMS (сучасних матеріалів та технологіях виготовлення MEMS) та компонентах і технологіях автоматизації видобутку, переробки та транспортування нафти та газу.

Редакційна колегія: І.Ш. Невлюдов, В.В. Євсєєв.

Manufacturing & Mechatronic Systems 2026: Proceedings of X st International Conference, Kharkiv, October 22-23, 2026: Thesises of Reports / [Ed. I.Sh. Nevlyudov (chief editor).] .- Kharkiv .: [electronic version], 2026. - 87 p.

The collection includes the thesises of reports on modern trends in the development of technologies and means of production and mechatronic systems, top experience and implementation of them in fields of: industrial automation and production management systems; systems engineering; CAD/CAM/CAE systems; mechatronics (electrical and mechanical systems, electronic control tools, mechanical CAD systems); robotics and intellectual toolls; MEMS (modern materials and manufacturing technologies MEMS) and components and technologies for the automation of oil, gas and oil extraction, processing and transportation.

Editorial board: Igor.Sh. Nevludov, Vladyslav.V. Yevsieiev

© Кафедра комп'ютерно-інтегрованих технологій, автоматизації, робототехніки та безпекової інженерії (КІТАРБІ), ХНУРЕ,2026

Міністерство освіти і науки України (МОНУ)
Харківський національний університет радіоелектроніки (ХНУРЕ)
Варшавський університет сільського господарства (WULS - SGGW)
Азербайджанський державний університет нафти і промисловості
Національний університет «Львівська політехніка»
Festo Didactic Україна
Jabil Circuit Ukraine Limited
ТОВ «Науково-виробниче підприємство «УКРІНТЕХ»»
Факультет автоматизації, комп'ютерно-інтегрованих технологій та систем (АКТС)
Кафедра комп'ютерно-інтегрованих технологій, автоматизації, робототехніки та безпекової інженерії (КІТАРБІ),
Державне підприємство «Харківський науково-дослідний інститут технології машинобудування»
Державне підприємство «Південний державний проектно-конструкторський та науково-дослідний інститут авіаційної промисловості»

МАТЕРІАЛИ

X-ої Міжнародної Конференції

ВИРОБНИЦТВО
&
МЕХАТРОННІ СИСТЕМИ 2026

(22-23 жовтня 2026)
Харків, Україна

ЗМІСТ

Інна Хондак

Інтелектуальні робототехнічні системи як засіб підвищення безпеки життєдіяльності та ефективності цивільного захисту …….. 10

Vladyslav Yevsieiv

Intelligent Method of Adaptive Neuro-PWM Speed Control for a Mobile Robot in a Constrained Workspace …… 13

І. В. Жарікова

Контактні пристрої на основі гнучких комутаційних структур для електричного контролю електронних компонентів…….. 17

Anton Andreiev, Svetlana Sotnik

Architecture of an Automatic Speech-Based Production Task Generation Service for Flexible Robotic Systems ……. 20

Y. V. Shalko

Conceptual Approaches to Enhancing the Intelligence of Navigation Methods for Mobile Warehouse Robots in Dynamic Environments ……. 25

Alina Konieva, Svetlana Sotnik

Methods of Terrain Traversability Assessment Based on Image Analysis for Mobile Robots …… 29

Дмитро Гурін

Методи ідентифікації оператора у робочій зоні колаборативного робота 34

Олександр Мамонтов

Вдосконалення методу CLM з урахуванням системного характеру небезпек виробничого процесу …….. 37

Diana Sukhomlinova, Svetlana Sotnik

Overview of Online Remote Control Architectures for Unmanned Aerial Vehicles …….. 41

Kyrylo Horovyi

Comparative Analysis of Wedge Bonding Methods and Factors Affecting the Quality of Contact Interconnections in Microelectronics …….. 46

Overview of Online Remote Control Architectures for Unmanned Aerial Vehicles

Diana Sukhomlinova, Svetlana Sotnik

Department CITARSE, Kharkiv National University of Radio Electronics, Ukraine,
Kharkiv, av. Nauki. 14., email: diana.sukhomlinova@nure.ua

Annotation: This work reviews and compares architectures of online remote control systems for unmanned aerial vehicles (UAVs), with a focus on swarm control. Centralized, decentralized, and hybrid (Edge-Cloud) architectures are analyzed in terms of fault tolerance, scalability, dependence on the communication channel, and ease of integration with a web interface. A hybrid architecture with a microservice server layer is substantiated as a rational choice: the server handles mission planning, task allocation, and operator interaction, while time-critical functions – formation control, local navigation, and obstacle avoidance – are executed onboard, which allows the swarm to keep operating when the server connection is lost. A software and hardware stack is proposed (React, Node.js, Python services, and companion computers running ROS 2), together with a distribution of protocols by communication segment: HTTP/REST and WebSocket for operator interaction, MQTT for server–vehicle exchange, WebRTC for video, and ROS 2/DDS for inter-vehicle communication. The paper also covers two-level navigation integration (global planning on the server and local obstacle avoidance onboard) and security measures for communication channels. Directions for future research include state synchronization between the server and onboard levels and experimental latency evaluation.

***Key words*:** control system architecture, UAV, swarm control, autonomous navigation, hybrid architecture.

I. INTRODUCTION

The rapid development of robotics and unmanned aerial vehicle (UAV) technologies creates an urgent need to build efficient, scalable, and reliable control systems [1-6]. Web-based systems have become particularly relevant in modern conditions, allowing for the continuous monitoring and control of UAVs from anywhere in the world [7-11]. This approach eliminates the need to install heavy specialized software on operator client devices, significantly increasing the mobility of system deployment.

In the context of the global transition from individual piloting to swarm management (swarm robotics), stringent requirements are placed on the architecture of such systems. It must ensure low network latency when transmitting control commands, minimal dependence on communication channels, high throughput for real-time telemetry aggregation, and flexible integration of complex autonomous navigation algorithms.

At the same time, the variety of modern approaches to building such systems – ranging from centralized client-server and cloud solutions to distributed architectures based on edge computing – complicates the choice of an optimal variant for specific operational conditions. The use of protocols and technologies such as WebSocket, WebRTC, MQTT, MAVLink, and ROS 2, along with microservice approaches to server-side organization, involves a series of trade-offs between data transfer latency, scalability, fault tolerance, and integration complexity. Additional challenges stem from communication link instability, cybersecurity requirements, and the need for continuous operation under partial connection loss.

The relevance of this topic is driven by the rapid growth in the scale of UAV applications in logistics, infrastructure monitoring, search and rescue operations, and the defense sector. This demands a transition from fragmented software solutions to unified, open, and scalable web control platforms. The lack of a unified approach to designing such systems, combined with a shortage of generalized comparative studies, makes it difficult for developers to select architectural solutions. This increases design risks, raises development costs, and reduces the reliability of the resulting systems. Therefore, systematizing existing architectures, analyzing their advantages and disadvantages, and identifying promising development directions hold significant theoretical and practical value.

The aim of this work is to review and conduct a comparative analysis of architectures for online remote UAV control systems, as well as to justify the choice of a hybrid (Edge-Cloud) microservice architecture. This proposed architecture combines web technologies for

II. ANALYSIS OF ARCHITECTURES AND TECHNOLOGIES FOR IMPLEMENTING ONLINE REMOTE CONTROL SYSTEMS FOR UNMANNED AERIAL VEHICLES

Based on the allocation of computing resources and decision-making centers, remote control system architectures are traditionally divided into centralized, decentralized, and hybrid (Edge-Cloud) models [8, 12].

In centralized systems, all calculations and decision-making are concentrated on a single server, with UAVs acting merely as execution devices.

Conversely, decentralized architectures rely on autonomous agents that interact directly with one another, typically within mesh networks, without depending on a central control node. Hybrid (Edge-Cloud) models combine both approaches: strategic functions (mission planning, task allocation, and route replanning) are executed at the server (cloud) level, whereas tactical functions (formation coordination, mutual collision avoidance, and obstacle response) are handled directly onboard.

The choice of a specific model directly affects the drone swarm's ability to execute missions under electronic interference or communication loss, and determines key system characteristics such as fault tolerance, scalability, communication link dependency, and ease of integration with the operator's web interface. A comparative analysis of these architectures is presented in Table 1.

Table 1. Comparison of online remote control system architectures

Architecture type	Computation distribution	Dependency on communication channel	Fault Tolerance	Scalability	Ease of Web Integration
Centralized	All calculations occur on a single server. UAVs act solely as executors.	High dependency: every command passes through the server; loss of connection results in loss of control.	Low (presence of a Single Point of Failure).	Low (presence of a Single Point of Failure).	High (single data aggregation point for the web client).
Decentralized	Each UAV is an autonomous agent (Mesh networks) [13].	Low dependency: decisions are made locally; data exchange occurs directly between vehicles.	High (failure of a node does not disrupt the swarm).	High in terms of computational load, but limited by mesh bandwidth: inter-node communication increases with node count, requiring interaction radius constraints.	Low (lack of a single hub for browser broadcasting).
Hybrid (Edge-Cloud)	Global planning on the server; local response on the UAV.	Moderate dependency: critical decisions are made on board; the server is not in the real-time control loop.	High (ability to complete the mission autonomously).	High, as computations are split between onboard systems and the server; the server processes telemetry and statistics, downsampling and aggregating them as needed for browser display, while video, as the heaviest stream, scales via an SFU (Selective Forwarding Unit) media server cluster.	Moderate (a single gateway exists for the web client, but some decisions are made on board, requiring state synchronization between the vehicle and the server).

The most balanced approach for modern web-based multi-UAV control systems is a hybrid (Edge-Cloud) architecture [14]. In this model, the server layer is responsible for global mission planning, task allocation among vehicles, user interaction via the interface, and overall telemetry aggregation. The onboard (Edge) layer handles time-critical processes: group formation control, local navigation, and dynamic obstacle avoidance; these run in real-time and operate independently of the server. However, the hybrid model also presents challenges: it requires state synchronization between the server and onboard layers, as well as a clear division between locally made and centralized decisions.

Practical implementation of a hybrid (Edge-Cloud) architecture demands a precise distribution of functional responsibilities among software components and the selection of technologies tailored to each layer's specifics. The server layer must ensure operator interaction, global planning, data aggregation, and video stream relaying, whereas the onboard layer provides real-time autonomous response and formation control. Therefore, the next step is analyzing the hardware and software stack required to realize this model as a cohesive, web-oriented system.

To build a reliable web-oriented control system using autonomous navigation algorithms, it is advisable to structure the server layer of the hybrid model using a modern microservice architecture. This allows isolating distinct logical domains and employing the most suitable technology stack for each specific task – from interface rendering to complex mathematical computations. The hardware foundation consists of a companion computer (such as NVIDIA Jetson or Raspberry Pi) executing local navigation; a flight controller running PX4 or ArduPilot firmware to manage the actuators; and wireless communication channels. The onboard computer connects to the server layer via Wi-Fi and LTE/5G, which provide sufficient bandwidth for video streaming, as well as IP-enabled radio modems suitable for telemetry and command transmission. Inter-vehicle communication is handled by a dedicated mesh channel (e.g., Wi-Fi mesh or specialized mesh radio modules). The onboard computer also functions as a gateway: it translates MAVLink messages into MQTT and vice versa, and relays the vehicle's state from ROS 2 to MQTT [15].

The presentation layer (frontend) is traditionally implemented as a Single Page Application (SPA). Leveraging

the component-based approach offered by React enables the creation of a high-performance reactive operator interface. Such an interface can render large volumes of geospatial data without page reloads, track UAV swarm movements on a digital map in real-time, and offer intuitive tools for visual patrol route setup. Vue or Angular serve as viable alternatives, though React offers a broader ecosystem of mapping libraries.

The server layer (backend) functions as an asynchronous communication core. Event-driven runtime environments like Node.js are well-suited for these tasks. Its non-blocking I/O architecture accommodates a high volume of concurrent persistent connections [16], which is critical for real-time routing of dense telemetry streams from vehicles to client browsers with minimal latency. Go represents another viable option due to its efficient handling of concurrent connections. Communication between the server and onboard systems is mediated by an MQTT broker, with flight history and telemetry persisted in a time-series database. For video transmission, the server layer additionally incorporates a signaling service, an SFU media server, and STUN/TURN servers for NAT traversal.

The intelligent data processing and global navigation layer is typically isolated into dedicated microservices. Because tasks such as optimal path planning through complex obstacles require matrix operations and graph theory, these modules are best developed in Python due to its rich ecosystem of algorithmic planning libraries (NumPy, SciPy, NetworkX). To interface with other services, these modules can be wrapped in REST endpoints using frameworks such as FastAPI. This structure allows calculating global trajectories for the entire UAV swarm prior to dispatching commands to the onboard computers.

III. NETWORK INTERACTION AND COMMUNICATION PROTOCOLS

The selection of network protocols in a web-oriented system is a decisive factor in ensuring the stability of a UAV swarm. Different data streams require different approaches to delivery reliability and speed [4, 17].

In a hybrid architecture, five communication segments can be identified: "operator–server", "server – drone", "drone – drone", "drone – server (SFU) – operator" (video stream), and internal onboard "onboard computer–flight controller", with each segment benefiting from specific protocols (Table 2). The interaction between the server layer and the UAV group is implemented via the lightweight MQTT protocol. Drones publish their coordinates and status updates to designated topics, while the server subscribes to them and publishes commands to individual drone topics. For high-frequency telemetry, QoS 0 is recommended, whereas QoS 1 or QoS 2 is suitable for critical commands. However, QoS 2 increases latency due to additional handshake acknowledgments; thus, under unstable channel conditions, QoS 1 with idempotent commands – where re-receiving a message does not trigger duplicate execution – is preferred. Since the broker is a central node in this exchange, deploying it as a cluster is recommended to eliminate a single point of failure [18]. On board, a gateway converts MAVLink messages from the flight controller into MQTT and vice versa, enabling the integration of standard flight controllers without modifying their firmware.

The video stream is transmitted via WebRTC through a media server (SFU) located at the server layer. Signaling for session establishment is performed over a WebSocket connection on the operator's side and via MQTT on the onboard computer's side.

Table 2. Communication protocols of a web-oriented UAV control system

Protocol	Exchange segment	Main purpose in the system	Latency level	Features and advantages
1	2	3	4	5
HTTP/REST	Operator – server	Authorization, loading history, mission setup.	Higher (request-response model)	Used for latency-insensitive operations.
WebSocket	Operator – server	Streaming telemetry to the operator's browser and WebRTC signaling.	Low	Full-duplex communication between the React front-end and the Node.js back-end; HTTP headers are not retransmitted once the connection is established.
MQTT	Server – onboard computer	Telemetry and mission commands; WebRTC signaling from the onboard computer.	Low	Publish/Subscribe pattern via a broker; QoS levels 0, 1, and 2 allow choosing a balance between speed and guaranteed delivery. Suitable for unstable wireless channels.

Table continuation 1

1	2	3	4	5
WebRTC	Onboard computer – server (SFU) – operator (video stream)	Streaming video from cameras (FPV/monitoring).	Low	Real-time video streaming (SRTP over UDP) directly to the operator's browser without installing additional software; requires a signaling server and STUN/TURN for NAT traversal. The media server (SFU) ingests a single stream from each vehicle and relays it to operators, relieving load from the onboard channel.
MAVLink	Onboard computer – flight controller	Control commands and telemetry for the flight controller (PX4, ArduPilot).	Minimal	Compact binary message format.
ROS 2 / DDS	Onboard – onboard	Exchanging state and maneuvers within the swarm.	Low	Decentralized node discovery and QoS tuning; operates without server connectivity.

Exchange between adjacent vehicles (coordinate updates, avoidance maneuvers) is best performed directly via ROS 2/DDS over a mesh network rather than through the server. This ensures that the formation is preserved even if connectivity to the server level is lost, adhering to hybrid architecture principles.

Because radio channels are vulnerable to interception and command spoofing, data exchange must be protected using encryption and integrity checks at the transport protocol layer (HTTPS for REST, WSS for WebSocket, MQTT over TLS, DTLS-SRTP for WebRTC, and SROS 2 for DDS), mutual device authentication (e.g., via client certificates), and role-based access control (RBAC) for operators [19-21]. The internal onboard MAVLink domain does not leave the vehicle; thus, it is secured through physical isolation and, if necessary, MAVLink 2 message signing.

IV. INTEGRATION OF AUTONOMOUS NAVIGATION ALGORITHMS

Swarm management typically requires a multi-level navigation system. High-level, or global, planning is performed on the server using a global static terrain map and encompasses two tasks. Task allocation among drones is executed using assignment algorithms, such as the Hungarian algorithm or auction-based methods [22]; cost matrices are generated based on estimated distances or path lengths to targets. For each vehicle's assigned goal, a route is generated using classical pathfinding algorithms (such as A* or Voronoi diagram-based methods). To prevent trajectory intersections, routes are separated by altitude or time (flight levels, time slots) or managed via multi-agent planning; residual conflicts are resolved by onboard local navigation.

Low-level, or local, navigation runs directly on the onboard computer of each UAV under ROS 2. It processes raw data from onboard sensors in real time to detect unpredictable dynamic obstacles (such as other aircraft or moving ground objects).

Local navigation algorithms, such as Artificial Potential Fields (APF) or Vector Field Histograms (VFH), allow the drone to bypass obstacles without waiting for new instructions from the central server. The potential field method calculates the direction of motion as the sum of attractive forces toward the target and repulsive forces from obstacles, while VFH builds a polar obstacle density histogram around the vehicle and selects the open direction closest to the goal. However, potential field methods are susceptible to local minima, making it advisable to combine them with a global planner that can re-route the path when a connection is available.

In the event of a lost connection to the server, local minimum escape is handled onboard via random trajectory perturbations, switching to a local planner using the onboard map, or, if further motion is impossible, executing an emergency procedure (holding position or returning to launch). Updated maneuver coordinates are broadcast to neighboring vehicles with minimal delay via ROS 2/DDS, preserving overall formation and preventing intra-swarm collisions.

To maintain formation, a Leader-Follower scheme may be employed. To prevent the leader from becoming a single point of failure, its role is dynamic: if heartbeat messages from the leader are absent for a specified interval, follower vehicles elect a new leader according to a defined priority (for example, by identifier; if battery level is taken into account, a hysteresis threshold is applied to prevent constant switching of the leader), while maintaining their positions relative to nearest neighbors until elections conclude. Because DDS lacks a native leader election mechanism, this logic is implemented at the application level.

In the event of a swarm partition into isolated subgroups, each subgroup can elect its own leader and act autonomously; once connectivity is restored, re-election under the same priority criteria consolidates management under a single leader. The onboard gateway of each vehicle relays its ROS 2 state to MQTT, thereby delivering the swarm status to the server level for re-routing as needed.

Thus, the server determines *what* each vehicle should do and *where* it should go, while real-time interaction during mission execution – including formation maintenance – remains at the onboard level.

V. CONCLUSIONS

The review shows that for web-oriented UAV swarm control systems, it is advisable to adopt a hybrid (Edge-Cloud) architecture with a microservice organization at the server level. A clear functional division – where the React interface, Node.js back-end, and Python planning service handle dispatching and operator interaction, while onboard computers running ROS 2 execute reactive autonomous navigation – creates the prerequisites for successful mission execution. Allocating protocols by communication segment (HTTP/REST and WebSocket for the operator, MQTT for server–vehicle communication, ROS 2/DDS for inter-vehicle exchange, and WebRTC for video) separates latency-sensitive processes from non-critical ones and maintains swarm operational integrity even upon loss of server connection. Key directions for future research include state synchronization between server and onboard levels, communication channel cybersecurity, and experimental evaluation of latency on a testbed or in simulation.

LIST OF REFERENCES

[1] I. Nevliudov, Sh. Omarov, and S. Sotnik, "Evolution of physical ai in humanoid robotics: from Atlas locomotion to Optimus neural networks," in *Proc. Int. Sci.-Pract. Conf. "Towards the Fifth Industrial Revolution – the Role of Universities, Challenges and Opportunities"*, 2026, pp. 393-395

[2] I. S. Nevliudov, et al., "Design of the structure and motion control system of a stationary robot manipulator for construction work," *Нові технології в будівництві*, no. 47, pp. 37-45, 2025

[3] S. V. Sotnik, "Support systems for robotics: principles, algorithms and development prospects," *Journal of natural sciences and technologies*, vol. 4, no. 2, pp. 419-430, 2025. DOI: 10.5281/zenodo.18088674

[4] D. A. Sukhomlinova, et al., "Analysis of main tasks and challenges in drone remote control," in *Proc. 18th Int. Sci.-Pract. Conf. "Information Technologies and Automation – 2025"*, Odesa, Ukraine, Oct. 30-31, 2025, pp. 1059-1061

[5] Y. Floru, et al., "Robotic process automation and integration systems for SMBS: priority processes and software comparison," in *Proc. 3rd All-Ukr. Conf. Computer-Integrated Tech., Automation Robotics (CITAR 2026)*, Kharkiv, Ukraine, May 14-15, 2026, pp. 11-15

[6] D. Sukhomlinova, et al., "Optimization of drone trajectory algorithms," in *Proc. 9th Int. Conf. Manufacturing & Mechatronic Systems (MMS 2025)*, Kharkiv, Ukraine, Oct. 25-26, 2025, pp. 46-49

[7] D. A. Sukhomlinova, et al., "Aerial robot in urban environments," in *Proc. 1st Int. Conf. Sustainable Smart Cities Communities: Business Innovation Solutions*, Kharkiv, Ukraine, Apr. 21, 2025, pp. 45-46

[8] M. S. Bielik, et al., "Analysis of the structures of mobile platforms for promoter robots," in *Proc. 5th Int. Sci.-Pract. Conf. "Global Trends in Science and Education"*, Kyiv, Ukraine, June 2-4, 2025, pp. 319-325

[9] T. A. Lykho and S. V. Sotnik, "Pattern recognition and computer vision technologies in decision support systems of robotic systems," in *Proc. 17th Int. Sci.-Pract. Conf. "Information Technologies and Automation – 2024"*, 2024, pp. 645-648

[10] A. S. Andreiev, et al., "Methods for improving the energy efficiency of small language models for autonomous robotics," in *Proc. 3rd All-Ukr. Conf. Computer-Integrated Tech., Automation Robotics (CITAR 2026)*, Kharkiv, Ukraine, May 14-15, 2026, pp. 6-10

[11] С. Сотник, "Розробка автоматизованої інформаційно-пошукової системи вибору маніпулятора промислових роботів," *Електромеханічні і енергозберігаючі системи*, 1 (68), pp. 52-58, 2025

[12] J. Stephan, J. Fink, A. et al., "System architectures for communication-aware multi-robot navigation," in *Proc. 2016 IEEE Int. Conf. Acoustics, Speech Signal Process. (ICASSP)*, Shanghai, China, 2016, pp. 6395-6399

[13] A. D. Yechevskyi, et al., "Research of orientation methods of autonomous mobile robots in industrial conditions," in *Proc. Conf. "Computer-Integrated Technologies, Automation and Robotics" (CITAR-2025)*, 2025, pp. 115-119

[14] J. Zhang, et al., "Distributed robotic systems in the edge-cloud continuum with ROS 2: A review on novel architectures and technology readiness," in *Proc. 2022 7th Int. Conf. Fog Mobile Edge Comput. (FMEC)*, Paris, France, 2022, pp. 1-8

[15] Y. Bielyi, et al., "Implementing a Secure Data Transmission Channel Based on Mesh VPN Under Limited IPv4 Addressing," *Int. J. Acad. Inf. Syst. Res. (IJAISR)*, vol. 10, no. 6, pp. 30-37, 2026

[16] P. Sooryavanshi, S. Upganlawar, et al., "Implementation of Node.js server on Raspberry Pi to control a remote vehicle for defense use," in *Proc. 2017 Int. Conf. Intell. Sustain. Syst. (ICISS)*, Palladam, India, 2017, pp. 816-819

[17] K. Mandrykov, et al., "Comparative Analysis of Industrial Data Transmission Protocols (IIOT) in Automation Systems," in *Proc. All-Ukr. Conf. "Intelligent Civil Safety Technologies and Robotic Systems for Emergency and Rescue Operations" (ICSTRO-2026)*, Kharkiv, Ukraine, Feb. 2026, pp. 49-53

[18] B. Bevilaqua, et al., "Self-Managed Federation of MQTT Brokers with Dynamic Topology Control," *J. Comput. Sci.*, vol. 19, pp. 1398-1409, 2023.

[19] R. Marunich, et al., "Security Analysis of Protocols for Integration With Access Control System," in *Proc. All-Ukr. Conf. "Intelligent Civil Safety Technologies and Robotic Systems for Emergency and Rescue Operations" (ICSTRO-2026)*, Kharkiv, Ukraine, Feb. 12-13, 2026, pp. 59-63

[20] T. Cherednichenko, et al., "Features of automatic working time control systems," in *Proc. 9th Int. Conf. "Manufacturing & Mechatronic Systems 2025"*, Kharkiv, Ukraine, Oct. 25-26, 2025, pp. 54-57

[21] S. Sotnik, "Performance and Simulation-Based Comparison of IoT Access Control Systems Using ESP32, Arduino Uno, and Raspberry Pi Pico W," *Appl. Cybersecurity Internet Governance*, 2026

[22] Y. Song, Z. Ma, N. Chen, S. Zhou, and S. Srigrarom, "Comparative analysis of centralized and distributed multi-UAV task allocation algorithms: A unified evaluation framework," *Drones*, vol. 9, no. 8, p. 530, 2025