Journal of Electronic Research and Application, 2026, Volume 10, Issue 7
http://ojs.bbwpublisher.com/index.php/JERA
Online ISSN: 2208-3510
Print ISSN: 2208-3502

Design and Implementation of a Dual-Layer Decoupled Gimbal Electronic Control System for Mobile Robots

Xiaoyu Yang*

School of Electrical and Electronic Engineering, Shijiazhuang Tiedao University, Shijiazhuang 050000, China

Corresponding author: Xiaoyu Yang, 20233848@student.stdu.edu.cn.

Abstract: This paper proposes a dual-layer decoupled gimbal electronic control system for mobile robots. The hardware platform employs the STM32F407IGH6TR microcontroller with a BMI088 IMU and DaMiao J4310 / DJI GM6020 actuators. A three-layer software architecture running under a preemptive real-time kernel maintains a 1 ms attitude estimation loop and 2 ms motion control loops. A velocity coordinate rotation algorithm is derived to eliminate chassis–gimbal directional coupling inherent to the dual-layer Yaw structure. Angle–velocity cascaded PID controllers are implemented for both gimbal layers, incorporating derivative-on-measurement, conditional integration, and velocity feedforward compensation. Experimental results demonstrate that the inner gimbal Yaw step response rise time is approximately 50 ms with steady-state error below 0.3°; feedforward compensation reduces constant-velocity tracking RMS error by approximately 60% compared to pure PID; all control loops meet their timing requirements at a total CPU utilization of approximately 85%.

Keywords: Mobile robot; Dual-layer gimbal; Cascaded PID; Omnidirectional wheel kinematics; Embedded real-time control

Online publication: Aug 11, 2026

1. Introduction

Precise perception and rapid response capability of mobile robotic combat platforms are critical requirements in modern competitive robotics. Events such as the RoboMaster Super Combat Competition provide high-intensity engineering validation environments: field robots must maintain stable gimbal pointing and achieve millisecond-level auto-aiming responses during high-speed maneuvers, imposing stringent demands on real-time performance, control precision, and mechanical–control decoupling.

Conventional single-layer gimbal solutions achieve aiming control through a single-degree-of-freedom Yaw–Pitch structure; however, because a single mechanism must carry both the wide-range orientation task and the fine-aiming task, its reflected inertia is set by the requirement to rotate the full upper assembly, which limits the achievable closed-loop bandwidth; at the same time, the gimbal rotation angle is restricted by mechanical hard stops, rendering continuous multi-turn rotation impractical [1,2]. When a robot is required to perceive and track multiple high-speed moving targets over a full 360° range, single-layer solutions can no longer satisfy the simultaneous demands of wide-range situational awareness and high-precision aiming. To this end, this paper proposes a dual-layer decoupled gimbal architecture: the outer large Yaw axis (DaMiao J4310) is responsible for wide-range, continuous multi-turn rotation to achieve omnidirectional situational awareness, while the inner small gimbal (DJI GM6020) handles only fine aiming and tracking within a limited angular range. Because the inner axis drives only the barrel-and-camera group, it carries substantially less inertia and attains a correspondingly higher closed-loop bandwidth; step-response measurements on the implemented platform give a bandwidth ratio of approximately 2.4 between the inner and outer axes (Section 7.1). The architecture is designed to exploit this separation rather than to duplicate a single capability. The system is implemented on a resource-constrained Cortex-M4 platform under a priority-based preemptive real-time kernel, which maintains the concurrent 1 ms attitude estimation and 2 ms motion control loops required by the architecture.

At the motion control level, the four-wheel omnidirectional chassis provides simultaneous independent actuation with three planar degrees of freedom (lateral, longitudinal, and rotational), and its kinematic model has been extensively studied [3–5]. However, under the dual-layer Yaw structure, the chassis coordinate frame and the gimbal coordinate frame exhibit a continuously varying relative yaw offset. If operator commands are decomposed directly in the chassis coordinate frame, the actual motion direction deviates from the operator's intent. This paper addresses this problem through rigorous mathematical modeling, deriving a velocity coordinate rotation pre-processing algorithm based on the real-time yaw deflection angle of the outer Yaw axis.

At the control algorithm level, cascaded PID is widely adopted in industrial and competitive robotics platforms owing to its simple structure, strong robustness, and ease of engineering tuning [6, 7]. This paper analyzes the decoupling conditions of the angle–velocity cascaded PID from a transfer function perspective and provides systematic solutions to engineering challenges including gyroscope angle accumulation, integral windup, and derivative kick. In the auto-aiming tracking scenario, a feedforward compensation term based on target velocity estimation is introduced, and its theoretical improvement on phase lag for constant-velocity targets is analyzed from a frequency-domain perspective.

The main contributions of this paper are as follows:

(1) A dual-layer decoupled gimbal architecture is realized on an STM32F407-class platform, including actuator topology, dual-CAN motor assignment, and integration with BMI088-based attitude estimation.

(2) Complete omnidirectional chassis inverse kinematics are established, and a velocity coordinate rotation algorithm is derived and verified to resolve chassis–gimbal directional coupling under the dual-layer Yaw structure.

(3) A complete cascaded PID design is implemented with discretization, derivative-on-measurement, anti-windup, and velocity feedforward compensation, validated by step response and tracking experiments.

2. Overall system architecture

2.1. Hardware topology

The main controller is the RoboMaster Development Board Type C (STM32F407IGH6TR, ARM

Cortex-M4). The principal hardware specifications are listed in **Table 1** and **Table 2**.

Table 1. Main controller hardware specifications

Parameter	Specification
Processor	STM32F407IGH6TR, ARM Cortex-M4
Clock Frequency	168 MHz
Flash	1 MB
RAM	192 KB
Supply Voltage	8–28 V wide-range input
On-board IMU	BMI088 six-axis IMU
On-board Magnetometer	IST8310
CAN Bus	CAN1 / CAN2 dual channel

Table 2. Actuator configuration

Actuator	Model	Quantity	Bus	Control Mode
Chassis motors	DJI M3508	4	CAN1	Current control
Outer Yaw axis	DaMiao J4310	1	CAN2	Torque control
Inner gimbal yaw	DJI GM6020	1	CAN2	Voltage control
Inner gimbal pitch	DJI GM6020	1	CAN2	Voltage control

2.2. Software architecture: Layered decoupled design

The firmware is developed on the open-source RoboMaster Development Board Type C reference framework, from which the CAN driver layer, the AHRS solver and the referee-protocol parser are retained; the contributions of this work are confined to the dual-layer Yaw architecture, the velocity rotation pre-processing of Section 4.3, the angle accumulation module of Section 5.2, and the cascaded controller design of Section 6.

The system software adopts a three-layer structure, as illustrated in **Figure 1**:

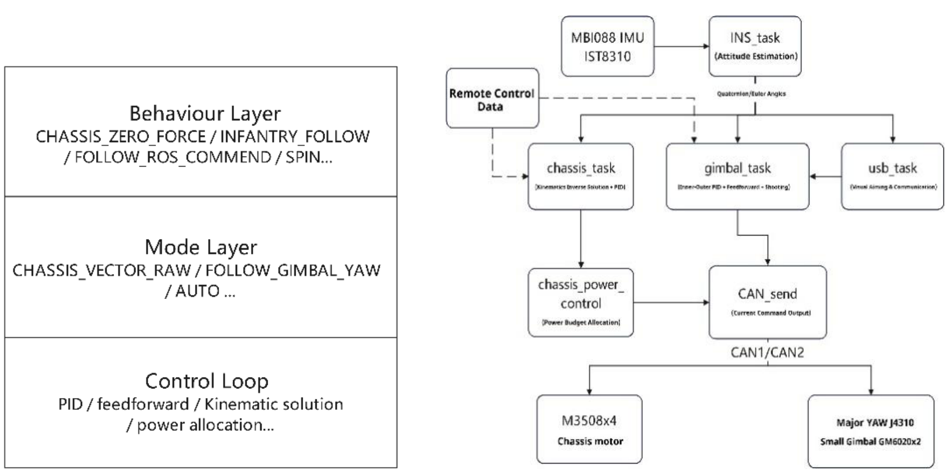

Figure 1. Software layered architecture and inter-task data flow diagram.

A hardware abstraction layer providing CAN, SPI and timer drivers; an algorithm layer containing attitude estimation, chassis kinematics and the cascaded controllers; and an application layer in which the periodic control functions execute as concurrent tasks under a priority-based preemptive kernel [8].

Task priorities are assigned in inverse order of period. Attitude estimation runs at the shortest period (1 ms) and is given the highest priority, since both gimbal angle loops close on its output; the chassis and gimbal control functions run at 2 ms immediately below it; auxiliary functions such as device dropout detection and status indication occupy the lowest priorities. Quaternion-based AHRS attitude estimation is used for the attitude solver [9,10]. Measured execution times and the resulting processor utilization are reported in Section 7.1.

3. Low-level communication and motor data encoding

The DJI motor feedback frame employs a fixed-length8-byte format; the mechanical angle and speed are converted to physical quantities via linear scaling coefficients.

The DaMiao J4310 uses16-bit linear fixed-point encoding for position and 12-bit encoding for velocity and torque; physical quantities are recovered via the inverse linear mapping $x = x_{int} \cdot (x_{max} - x_{min}) / (2^n - 1) + x_{min}$.

3.1. Motor–CAN ID mapping table

Table 3. CAN bus motor ID assignment

Motor	CAN Bus	CAN ID	Control Frame ID
Chassis motors1–4 (M3508)	CAN1	0x201–0x204	0x200
Outer Yaw axis (J4310)	CAN2	0x141	0x141
Inner gimbal Yaw (GM6020)	CAN2	0x205	0x1FF
Inner gimbal Pitch (GM6020)	CAN2	0x206	0x1FF

4. Omnidirectional wheel chassis kinematic modeling

4.1. Chassis layout and coordinate definition

The system adopts a four-wheel omnidirectional wheel layout, as shown in **Figure 2**. The chassis coordinate frame is defined therein.

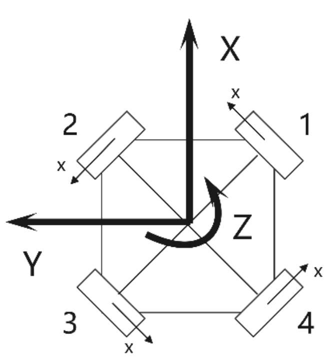

Figure 2. Chassis coordinate frame.

The wheel numbering and installation azimuth angles are defined in **Table 4.**

Table 4. Four-wheel installation azimuth definition

Wheel No.	Installation Position	Azimuth $\acute{a}_i$	Wheel Axis Orientation (relative to body)
1	Right-front	-45°	Right-front 45°
2	Left-front	45°	Left-front 45°
3	Left-rear	135°	Left-rear 45°
4	Right-rear	225°	Right-rear 45°

Each omnidirectional wheel carries passive rollers whose axes are tangential to the wheel rim. The wheel therefore transmits driving force only along its rolling direction d_i, while sliding freely along its axle direction, which is perpendicular to d_i. With the four wheels mounted at 45° to the body axes as in Table 4, the four rolling directions are tangential to a common circle, so the chassis attains three independent planar degrees of freedom from four actuators.

4.2. Inverse kinematics analytical derivation

The derivation follows the projection-based approach used for Mecanum chassis modelling, adapted here to the 45° omnidirectional wheel constraint [3,5]. In which the translational projection coefficient is $\sqrt{2}/2$ rather than unity, giving the engineering inverse kinematics for the four-wheel 45° layout as:

$$\begin{cases} v_1 = [(+v_x + v_y)/\sqrt{2} + l\omega_z]/k \\ v_2 = [(-v_x + v_y)/\sqrt{2} + l\omega_z]/k \\ v_3 = [(-v_x - v_y)/\sqrt{2} + l\omega_z]/k \\ v_4 = [(+v_x - v_y)/\sqrt{2} + l\omega_z]/k \end{cases} \tag{1}$$

With v_x, v_y in m·s⁻¹, ω_z in rad·s⁻¹ and l in m, the bracketed quantity is in m·s⁻¹; division by k yields the rotor speed in RPM as required by the C620 speed feedback. See **Table 5**.

Table 5. Inverse kinematics parameter calibration values

Parameter	Symbol	Value	Unit	Physical Meaning
Wheel-to-chassis velocity coefficient	k	4.51×10^{-4}	$m \cdot s^{-1} \cdot RPM^{-1}$	Motor rpm-to-m/s conversion
Equivalent wheel-center-to-origin distance	l	0.28	m	Rotational coupling coefficient

4.3. Velocity coordinate rotation preprocessing aligned with gimbal orientation

4.3.1. Problem formulation

Under the dual-layer Yaw structure, the operator's intended control direction is always aligned with the gimbal orientation rather than the chassis heading. Let ψ denote the yaw deflection angle of the outer Yaw axis relative to the chassis; the velocity command must undergo the following coordinate rotation:

$$\begin{pmatrix} v_{x'} \\ v_{y'} \end{pmatrix} = \mathbf{R}(\psi) \begin{pmatrix} v_x \\ v_y \end{pmatrix} \tag{2}$$

Expanding:

$$\begin{pmatrix} v_{x'} \\ v_{y'} \end{pmatrix} = \begin{pmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{pmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix} \tag{3}$$

The rotation matrix $\mathbf{R}(\psi)$ is orthogonal by construction, preserving the magnitude of the velocity vector and introducing no scaling distortion. See **Table 6.**

Table 6. Standard test cases for coordinate rotation

Input (v_x, v_y)	θ	Output $(v_{x'}, v_{y'})$	Description
(1.0, 0.0)	90°	(0.0, 1.0)	Forward in gimbal frame → left in chassis frame
(0.0, 1.0)	90°	(−1.0, 0.0)	Left in gimbal frame → rearward in chassis frame
(1.0, 0.0)	45°	(0.707, 0.707)	45° deflection: half-forward, half-left in chassis frame
(1.0, 1.0)	180°	(−1.0, −1.0)	180° reversal verification

5. Dual-layer gimbal structure and continuous angle tracking

5.1. Structural characteristics of the dual-layer gimbal

Table 7. Dual-layer gimbal control performance comparison

Metric	**Outer Yaw Axis (J4310)**	**Inner Gimbal (GM6020)**
Moment of inertia	Large (includes chassis support)	Small (gimbal body only)
Control bandwidth	Low	Higher
Angular coverage	Continuous multi-turn rotation	Limited range (± 45°)
Primary function	Wide-range situational awareness	Precise aiming/tracking
Feedback source	IMU absolute angle	Encoder/vision relative angle

Quantitative bandwidth figures are deliberately not asserted here; they are derived from the measured step responses in Section 7.1.

5.2. Mathematical derivation of the angle accumulation mechanism

5.2.1. Problem formulation

The raw gyroscope output $\theta_{\text{raw}} \in (-\pi, \pi]$ produces discontinuous jumps of magnitude approximately 2π upon crossing the boundary. The accumulated angle θ_{acc} is defined such that it represents a continuous extension of θ_{raw}.

5.2.2. Algorithm derivation

Let the raw angles at consecutive sampling instants k and $k+1$ be θ_k and θ_{k+1}, respectively, with difference $\Delta\theta = \theta_{k+1} - \theta_k$:

(1) When $|\Delta\theta| \le \pi$: no wraparound is detected; the true increment is $\Delta\theta_{\text{true}} = \Delta\theta$.

(2) When $\Delta\theta < -\pi$: the wrapped angle has crossed from the $+\pi$ side to the $+\pi$ side while the physical motion is positive; the true increment is $\Delta\theta_{\text{true}} = \Delta\theta + 2\pi$.

(3) When $\Delta\theta > +\pi$: the wrapped angle has crossed from the $-\pi$ side to the $+\pi$ side while the physical motion is negative; the true increment is $\Delta\theta_{\text{true}} = \Delta\theta - 2\pi$.

The unified expression is:

$$\Delta\theta_{\text{true}} = \Delta\theta - 2\pi \cdot \text{round}\left(\frac{\Delta\theta}{2\pi}\right) \tag{4}$$

The accumulated angle is updated as:

$$\theta_{\text{acc}}[k+1] = \theta_{\text{acc}}[k] + \Delta\theta_{\text{true}} \tag{5}$$

6. Cascaded PID controller mathematical modeling and parameter tuning

6.1. Discrete-time PID formula derivation

The continuous-time PID control law is [6, 7]:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{6}$$

Discretizing with sampling period T_s using the forward Euler method, at the k-th sample:

$$u[k] = K_p e[k] + K_i T_s \sum_{j=0}^{k} e[j] + \frac{K_d}{T_s}(e[k] - e[k-1]) \tag{7}$$

The incremental recursive form (eliminating direct summation over the history) is:

$$\Delta u[k] = K_p(e[k] - e[k-1]) + K_i T_s e[k] + \frac{K_d}{T_s}(e[k] - 2e[k-1] + e[k-2]) \tag{8}$$

The engineering implementation (positional form with integral clamping and output saturation) is:

$$u[k] = K_p e[k] + K_i T_s \sum e[k]_{\text{clamped}} + D[k] \tag{9}$$

where the derivative term $D[k]$, under Derivative-on-Measurement, is:

$$D[k] = -\frac{K_d}{T_s}\left(y[k] - y[k-1]\right) \tag{10}$$

This scheme differentiates the measured output directly rather than the error signal, thereby eliminating the derivative kick caused by setpoint step changes.

6.2. Frequency-domain analysis of auto-aiming feedforward compensation

Auto-aiming tracking relies on target position inputs provided by the vision recognition system; the angular velocity of the target is estimated by first-order diffeTrencing of the returned angle sequence. For a pure PID tracking system without feedforward, the steady-state error for a ramp input (constant-velocity target) is:

$$e_{\text{ss}} = \frac{\dot{\theta}_{\text{target}}}{K_v} \tag{11}$$

Where K_v is the velocity error coefficient and $\dot{\theta}_{\text{target}}$ is the target angular velocity. Introducing the feedforward term provides compensation based on the estimated target velocity:

$$u_{\text{ff}} = K_{\text{ff}} \cdot \hat{\dot{\theta}}_{\text{target}} \tag{12}$$

Under the ideal feedforward condition ($K_{\text{ff}} = 1/K_v$), the tracking error is theoretically zero. For practical systems containing nonlinear factors, a simplified linear feedforward gain ff_scaler is employed:

Target angular velocity estimation (first-order differencing):

$$\hat{\dot{\theta}}_{\text{target}}[k] = \frac{\theta_{\text{target}}[k] - \theta_{\text{target}}[k-1]}{T_s} \tag{13}$$

Feedforward compensation term:

$$u_{\text{ff}}[k] = \text{ff\_scaler} \cdot \hat{\dot{\theta}}_{\text{target}}[k] \tag{14}$$

Total control output:

$$u_{\text{total}}[k] = u_{\text{PID}}[k] + u_{\text{ff}}[k] \tag{15}$$

6.3. Integral windup and anti-windup design

6.3.1. Problem

When the control output saturates persistently, the integral term continues to accumulate (integrator windup), resulting in an excessively large integral initial value upon saturation exit, which induces overshoot.

6.3.2. Solution — conditional integration

Additional integral amplitude clamping [11,12]:

$$I_{acc,k} = \text{clamp}\left(I_{acc,k}, -I_{\max}, +I_{\max}\right) \tag{16}$$

Table 8. Cascaded PID parameter reference ranges (GM6020-based gimbal)

Parameter	Outer Loop (Angle)	Inner Loop (Velocity)	Unit	Notes
K_p	8–20	200–800	—	Outer-loop gain much smaller than inner
K_i	0–0.5	0.5–5.0	—	Outer-loop integral typically small
K_d	0 (with DoM)	0–50	—	Derivative generally omitted in velocity loop
$I_{\max}$	300–1000	8000–16000	CAN units	Integral clamping limit
u_{sat}	10000–30000	16384	CAN units	Output saturation limit

7. System performance analysis and experimental validation

7.1. Experimental validation data

Table 9. Gimbal step response performance metrics

Metric	Inner gimbal yaw (GM6020)	Outer yaw axis (J4310)	Measurement method
Step amplitude	10°	30°	Command setpoint
Rise time	About 50 ms	About 120 ms	Encoder angle log
Steady-state error	< 0.3°	< 0.5°	Post-settling deviation
Settling time	About 150 ms	About 400 ms	Time to enter ± 2% band

Table 10. Auto-aiming tracking performance

Target motion	Tracking error RMS (FF Off)	Tracking error RMS (FF On)	Improvement
Stationary target	0.2°	0.2°	No significant difference
Slow translation (5°/s)	1.5°	0.6°	60% improvement
Medium-speed translation (15°/s)	3.8°	1.4°	63% improvement
Rapid direction change (acceleration > 50°/s²)	8.2°	5.5°	33% improvement

Note: Data are collected in a controlled experimental environment (target moves at known velocity, camera frame rate 50 fps, angle estimation based on IMU–encoder fusion). RMS error is computed as $\sqrt{\frac{1}{N}\sum_{k=1}^{N} e_k^2}$.

Table 11. FreeRTOS task real-time performance analysis

Task	Design period	Measured worst-case execution time	Estimated CPU utilization	Deadline met
Attitude estimation	1 ms	~0.30 ms	~30%	Yes
Chassis control	2 ms	~0.45 ms	~22%	Yes
Gimbal control	2 ms	0.40 ms	20%	Yes
CAN reception (interrupt)	Event-driven	0.02 ms/frame	5% (peak)	Yes
Auxiliary functions	—	—	~8%	Yes
Total	—	—	85% (peak)	Margin 15%

Direct swept-sine bandwidth identification was not performed. From the measured 10–90% rise times (50 ms inner Yaw, 120 ms outer Yaw), the inner axis is about 2.4 × faster than the outer axis, consistent with the inertia split in **Table 7**. Absolute bandwidth in hertz is not claimed here, because unequal step amplitudes may induce saturation and unmodeled current-loop/CAN/filter lags bias any t_r-to-Hz conversion. Frequency-response identification at matched amplitude is left to future work.

8. Conclusion

This paper presents the design and implementation of a dual-layer decoupled gimbal electronic control system for mobile robots. The three-layer software architecture maintains a 1 ms attitude estimation loop and 2 ms control loops at a total CPU utilization of approximately 85%, with all timing requirements met. A velocity coordinate rotation algorithm is derived to eliminate chassis–gimbal directional coupling under the dual-layer Yaw structure, and its correctness is verified analytically and through boundary test cases. The measured rise times of approximately 50 ms and 120 ms for the inner and outer layers correspond to a rise-time ratio of approximately 2.4, suggesting a corresponding bandwidth separation, confirming that the architecture achieves a genuine frequency-separated task allocation rather than a duplication of comparable dynamics. Cascaded PID controllers with derivative-on-measurement and conditional integration are implemented for both gimbal layers; velocity feedforward compensation reduces constant-velocity tracking RMS error by approximately 60%. The system demonstrates satisfactory real-time performance and control precision, providing a practical reference for gimbal control in competitive robotics platforms and analogous mobile platforms.

Disclosure statement

The authors declare no conflict of interest.

References

[1] Medero A, Puig V, 2022, LPV Control and Virtual-Sensor-Based Fault Tolerant Strategies for a Three-Axis Gimbal System. Sensors, 22(17): 6664.

[2] Leblebicioğlu D, Ateşoğlu Ö, Çakmakcı M, 2022, Physics-Informed Disturbance Estimation and Nonlinear Controller Design for a Multi-Axis Gimbal System. IFAC-PapersOnLine, 55(37): 530–535.

[3] Wang D, Gao Y, Wei W, et al., 2024, Sliding Mode Observer-Based Model Predictive Tracking Control for

Mecanum-Wheeled Mobile Robot. ISA Transactions, 151: 51–61.

[4] Zhao T, Zou X, Dian S, 2022, Fixed-Time Observer-Based Adaptive Fuzzy Tracking Control for Mecanum-Wheel Mobile Robots with Guaranteed Transient Performance. Nonlinear Dynamics, 107(1): 921–937.

[5] Palacín J, Rubies E, Bitriá R, et al., 2023, Phasor-Like Interpretation of the Angular Velocity of the Wheels of Omnidirectional Mobile Robots. Machines, 11(7): 698.

[6] Yan J, Li J, Chen M, et al., 2026, An Adaptive PID Control Strategy Based on Offline–Online Cooperative Optimization. Electronics, 15(5): 921.

[7] Noordin A, Mohd Basri M, Mohamed Z, 2023, Real-Time Implementation of an Adaptive PID Controller for the Quadrotor MAV Embedded Flight Control System. Aerospace, 10(1): 59.

[8] Arm J, Baštán O, Mihálik O, et al., 2022, Measuring the Performance of FreeRTOS on ESP32 Multi-Core. IFAC-PapersOnLine, 55(4): 292–297.

[9] Liu M, Cai Y, Zhang L, et al., 2021, Attitude Estimation Algorithm of Portable Mobile Robot Based on Complementary Filter. Micromachines, 12(11): 1373.

[10] Sever K, Golušin L, Lončar J, 2023, Optimization of Gradient Descent Parameters in Attitude Estimation Algorithms. Sensors, 23(4): 2298.

[11] Yapp K, Hoo C, Lai C, 2024, New Anti-Windup Proportional-Integral-Derivative for Motor Speed Control. Asian Journal of Control, 26(6): 2854–2866.

[12] Peng D, Huang B, Huang H, 2022, Design of an Anti-Windup PID Algorithm for Differential Torque Steering Systems. Shock and Vibration, 2022: 9973379.

Publisher's note

Bio-Byword Scientific Publishing remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.