Design and Integrated Implementation of a Unified Architecture for Intelligent Unmanned Ground Vehicles Based on MATLAB

Zhanxin Ye
School of Mechanical Engineering, UNSW, Mechatronic Engineering, NSW, 2052, Australia

Abstract: This paper addresses the challenges of complex hardware integration, fragmented software frameworks, and inconsistent interactions encountered during the development of unmanned ground vehicles (UGVs). It proposes and implements an intelligent UGV integrated control system centred on the MATLAB/Simulink platform. The system adopts a layered architecture comprising 'remote monitoring-onboard fusion-real-time control-sensor-based execution', integrating modules such as LiDAR terrain perception, GNSS/IMU tight-coupled positioning, Galil motion control, and Xbox remote operation. Through designing a unified multi-layer communication protocol and secure arbitration mechanism, and utilising MATLAB to complete the entire integration process from algorithm development and simulation validation to real-time monitoring, the system demonstrates stable module coordination and unimpeded closed-loop control command transmission in simulation experiments. This validates the effectiveness and engineering practicality of the proposed integrated solution, providing a replicable approach for rapid UGV prototyping.

Keywords: Matlab; UGVs; GNSS; IMU; Galileo

1. Introduction

Unmanned ground vehicles (UGVs), as a vital branch within the field of robotics and autonomous systems, demonstrate irreplaceable value in complex scenarios such as military reconnaissance, industrial inspection, agricultural automation, and disaster relief. A fully functional UGV system is not merely an integration of mechanical components, but rather a profound fusion of multiple technological modules including environmental perception, real-time decision-making, precise control, and human-machine interaction. A key challenge in advancing UGVs from laboratory prototypes to practical applications lies in efficiently and reliably integrating multi-source, heterogeneous hardware sensors and actuators, while establishing a unified, flexible software control and monitoring platform.

Within the field of multi-sensor fusion, existing research primarily focuses on algorithmic optimisation: Reference architectures and MBSE have been used to improve modularity and interoperability in autonomous ground vehicle systems [1]. Cai et al. (2023) proposed a multi-step joint filtering model based on Kalman filtering for low-cost multi-robot collaborative localisation [2]. Liu et al. (2022) advanced beyond traditional point-level fusion by introducing the BEVFusion general framework, which unifies multimodal features into a bird's-eye view (BEV) representation space [3]. Qin et al. (2023) introduced a supervised learning strategy named SupFusion, enhancing fusion network performance through feature-level supervision [4]. Pang et al. (2020) developed CLOCs, a lightweight post-fusion method operating at the detection candidate box level [5]. However, these works primarily focus on perception algorithms themselves, lacking system-level solutions for heterogeneous hardware integration and rapid engineering validation. Consequently, this paper endeavours to construct a unified development and deployment platform centred on MATLAB to bridge the gap between advanced algorithms and stable engineering applications.

2. Hardware System

The hardware system of the intelligent UGV provides the physical foundation for environmental perception, precise control, and human-machine interaction. This chapter details the hardware architecture of the platform, the rationale for selecting each subsystem, integration methodologies, and key interface designs. The system adopts a layered modular architecture comprising the 'remote monitoring

Copyright @ STEMM Institute Press

layer-onboard fusion layer-real-time control layer-sensing and execution layer', as illustrated, ensuring clear functional separation and integration flexibility.

2.1 Perception Subsystem: Environmental Awareness and Self-Localisation

The perception subsystem is responsible for acquiring terrain information and determining the vehicle's own position and orientation, thereby providing the basis for decision-making.

2.1.1 Lidar terrain perception module

To obtain real-time terrain profile data ahead of the vehicle, this system employs the SICK LMS151[6] two-dimensional laser scanner as its core terrain perception sensor. Operating on the time-of-flight principle, this radar scans within a 270° horizontal field of view at 0.5° angular resolution, with a maximum detection range of 50 metres and a measurement frequency of 50 Hz. Its high precision and reliability render it exceptionally suitable for terrain contour detection and obstacle recognition in outdoor unmanned ground vehicles (UGVs). Mounted via a protective bracket above the vehicle's front bumper, the scanner's plane remains parallel to the ground to optimise capture of road surface undulations and obstacles ahead.

Via an Ethernet interface, the radar transmits continuous polar coordinate scan data to the onboard computer using TCP protocol. Data processing occurs in real-time within the onboard computer, encompassing data preprocessing, ground point segmentation, terrain generation, and feature extraction.

In practical operation, the LMS151 first performs distance-validated data verification and intensity filtering on raw scan data to eliminate noise points caused by rain, fog, dust, or specular reflections. Polar coordinate points are subsequently transformed into a radar-centred vehicle coordinate system. Ground points are segmented from individual scan lines using an algorithm based on height thresholds and continuous slope constraints. By accumulating data from multiple consecutive frames and synchronously fusing it with vehicle pose information from GNSS/IMU, a two-dimensional elevation map of the area ahead of the vehicle is constructed. The final elevation map is generated, and key terrain features are calculated, such as the average gradient of the forward path, lateral slope, and navigable width. These features are extracted in real-time and formatted into a compact digital descriptor, transmitted to the control layer to provide direct input for vehicle speed and steering decisions.

2.1.2 GNSS/IMU integrated positioning module

To achieve high-precision, high-reliability self-localisation, this system employs a tightly integrated navigation system comprising a NovAtel PwrPak7 GNSS receiver and a MicroStrain 3DM-GX5-45 IMU. The GNSS antenna is mounted on an unobstructed section of the vehicle roof, while the IMU is rigidly installed near the vehicle's centre of mass, with its axes strictly aligned to the vehicle coordinate system.

The GNSS provides absolute position and velocity information expressed in latitude, longitude, and altitude, updated at a frequency of 10 Hz. The IMU delivers 200 Hz triaxial acceleration and angular velocity data. To address the susceptibility of GNSS signals to interference near trees or buildings, as well as the significant drift inherent in pure inertial navigation, the system employs an Extended Kalman Filter (EKF) running on the vehicle computer to achieve tight GNSS/IMU integrated navigation. This filter performs deep fusion using IMU data as the state prediction basis and GNSS raw pseudorange, carrier phase, and Doppler shift as observations. It ultimately outputs stable, continuous six-degree-of-freedom pose estimates at a high frequency of 200 Hz, encompassing position (UTM coordinates), velocity, attitude angles (roll, pitch, yaw), and position uncertainty covariance. This fused data forms the foundation for path tracking and global map construction.

2.2 Control and Drive Subsystem: Motion Actuators

This subsystem is responsible for precisely translating digital commands into physical motion.

2.2.1 Motion control core: galil controller

The Galil DMC-4183 four-axis Ethernet motion controller serves as the underlying motion control core. Its core value lies in providing hardware-level deterministic real-time control, entirely independent of the host computer's operating system. This ensures stable motor control loops even under the most severe communication delays. The controller connects to the onboard computer via Ethernet, receiving ASCII-formatted commands.

2.2.2 Drive and actuator systems

The drive system comprises two independent components:

Hub Motor Drive: The left and right rear wheels are directly driven by Maxon EC-i 40 brushless DC hub motors, each paired with an ESCON 70/10 servo drive. The drives operate in 'analogue voltage speed mode'. The Galil controller's two analogue output channels (AO0, AO1) output ±10V voltage signals directly corresponding to the motor's target speed. The motor's integrated 1000-line incremental encoder feeds pulse signals back to the Galil's digital input channels, forming a complete closed-loop system.

Steering servo control: Front-wheel steering is driven by a Hitec HS-7950TH high-torque digital servo. The Galil's third analogue output (AO2) connects to the servo controller, where the output voltage signal (0-3.3 V) linearly maps to the servo's steering angle range (-30° to +30°).

2.2.3 Closed-Loop control strategy

For velocity control, Galil incorporates an independent digital PID controller for each motor axis internally. The control law is as follows

$$u(t)=K_y e(t)+K_i \int_0^t e(\tau)d\tau+K_a \quad (1)$$

Where e(t) represents the error between the target rotational speed and the encoder feedback speed. Manual tuning of the PID parameters was performed using Galil's WSDK software, resulting in final values of Kp = 2.5, Ki = 0.8, and Kd = 0.1. This configuration enables the system to reach the target speed within 0.3 seconds without overshoot. Recent studies have shown that MPC-based approaches can improve UGV trajectory tracking performance under complex scenarios [7].

2.3 Human-Machine Interface and On-Board Computing Subsystem

This subsystem handles computation, communication, and interaction.

2.3.1 On-Board computing platform

The Intel NUC11 Performance Kit serves as the primary on-board computer, equipped with an i7 processor and 16GB RAM, running Ubuntu 20.04 LTS. It hosts all advanced algorithms, concurrently executing the following tasks: point cloud processing, EKF fusion filtering, communication with the host computer, and transmission of motion commands to Galil.

2.3.2 Wireless remote control interface

The Microsoft Xbox Wireless Controller serves as the primary remote control device. The controller connects to the remote host computer (laptop) via a USB wireless adapter, rather than the onboard computer. This design positions operator commands at the top decision-making layer, facilitating flexible integration and safe arbitration (e.g., emergency stop priority) between manual commands and autonomous decision commands within MATLAB.

3. Software Systems and Communication Protocols

MATLAB/Simulink has been adopted for co-simulation and virtual testing of autonomous rovers, supporting its use as an integrated development and validation environment (Martelli et al., 2025).[8] This chapter will detail the software architecture underpinning the entire system, the core algorithm development environment, the multi-threaded communication framework, and the design of key data protocols. This system innovatively employs MATLAB/Simulink as a unified top-level software framework, integrating end-to-end functionality from sensor data fusion and control algorithm design to human-machine interaction and real-time monitoring. An efficient, reliable multi-layer communication protocol network enables seamless data flow from decision-making to execution.

3.1 Overall Software Architecture and MATLAB's Core Role

The system's software architecture employs a layered design to maximise modularity and scalability, centred around the MATLAB ecosystem. MATLAB fulfils three core roles within this system:

Firstly, MATLAB constitutes the top-level control algorithm development and simulation environment. All core algorithms-including multi-sensor data parsing, adaptive control laws based on real-time terrain features, teleoperation command fusion logic, and path trackers-are developed and validated through graphical block diagrams within Simulink. This visual development approach significantly enhances prototyping efficiency. Crucially, the Simulink Coder toolchain automatically generates efficient, portable C++ code from thoroughly validated algorithmic models. Following cross-compilation, this code can be directly deployed to run on the vehicle's embedded computer, achieving seamless transition from offline simulation to online field deployment and establishing a complete rapid prototyping development chain.

Secondly, the MATLAB platform serves as an integrated real-time data visualisation and monitoring platform. Using MATLAB App Designer, we developed a dedicated graphical user interface for this system, functioning as the operator's 'virtual cockpit'. This monitoring interface integrates the comprehensive display of multi-source information: Firstly, on a two-dimensional map, it overlays in real time the vehicle trajectory (indicated by a red line) fused from GNSS/IMU data, pre-set or dynamically planned global/local paths (blue lines), and the vehicle's current position with precise heading arrows; Secondly, the dashboard panel provides real-time, dynamically updated displays of key vehicle states-including speed, steering angle, control mode, and battery voltage-presented both numerically and via analogue gauges. This highly integrated visual interface delivers comprehensive situational awareness to remote operators.

Finally, MATLAB scripts serve as the central communications and task management hub running on the host computer (remote laptop), orchestrating the entire system. This master's programme is responsible for establishing network connexions with all nodes, including the onboard platform and Galil controller, managing concurrent data transmission threads, and synchronously logging all experimental data. It also serves as the system-level entry point for Xbox wireless controller commands. Consequently, this hub effectively functions as the command dissemination centre and data aggregation point for the entire distributed system, ensuring orderly and reliable interaction of command and data flows between subsystems.

3.2 Data Synchronisation and Error Handling Mechanisms

To ensure stable system operation in complex environments, two core mechanisms have been implemented at the software level: data synchronisation and reliability assurance mechanisms. These enhance overall robustness by addressing information accuracy and system security respectively.

At the data synchronisation level, the system employs a fusion strategy based on high-precision timestamps to address challenges arising from asynchronous sampling across multiple sensors. All sensor data-including LiDAR, GNSS, and IMU-is stamped with microsecond-accurate timestamps at the source of acquisition. During Extended Kalman Filter (EKF) state estimation and terrain feature extraction on the onboard computing platform, processing modules employ interpolation or caching alignment techniques to synchronise perception data from different time points onto a unified decision-making time baseline. This ensures strict temporal consistency for all environmental and state information used in vehicle control, establishing a reliable data foundation for subsequent precise decision-making.

Regarding reliability assurance, the system incorporates multi-layered monitoring and arbitration strategies. Firstly, communication links employ 'heartbeat packets' and hardware watchdogs for health monitoring: The remote monitoring layer and the onboard fusion layer maintain heartbeat communications at a rate of once per second. Should no response be received within three seconds, the host monitoring interface triggers a 'connexion lost' alert and automatically issues a stop command. Concurrently, the Galil motion controller incorporates a hardware watchdog; if no new motion commands are received within its preset timeframe, it autonomously cuts motor output, achieving hardware-level safety shutdown. Furthermore, a safety arbiter is established at the top layer of the control logic. This component comprehensively processes remote operation commands from the Xbox controller, generated commands from the autonomous decision-making module, emergency stop signals, and system health status. It adjudicates based on a fixed priority hierarchy: 'Emergency Stop > Manual Override > Automatic Command'. Ultimately, it outputs an optimal control command that ensures safety, thereby constructing a comprehensive safety protection system spanning from communication links to decision-making logic.

4. Experiments and Results Analysis

4.1 Experimental Setup

The experiment was conducted on the established simulator. The test platform comprised the integrated UGV described in Chapter 2.

4.2 Subsystem Function Verification

During simulation testing, the GNSS/IMU tight integration navigation system operated stably. As demonstrated in Figure 1 and Figure 3, GNSS data decoding was correct (CRC checksum passed) and continuously outputted high-precision positioning information (with elevation displayed as 108.389 metres). This system provided the vehicle with a reliable reference for its position and orientation. Concurrently, as shown in Figure 2, the terrain processing algorithm for the two-dimensional LiDAR was successfully triggered and executed within the simulation environment, enabling real-time output of forward terrain features.

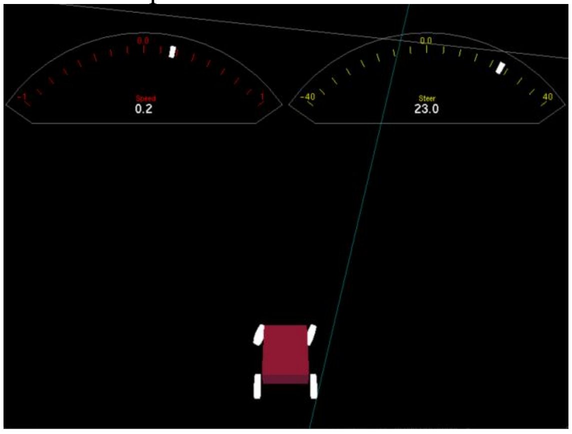


Figure 1. Figure of Simulator that Shows how the Car Runs

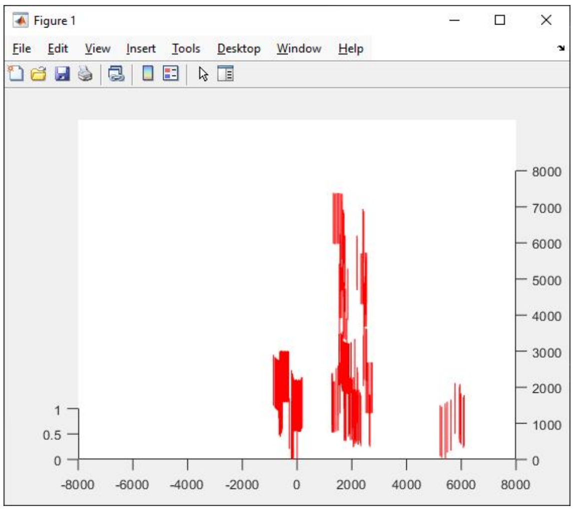
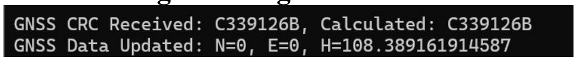


Figure 2. Figure of LiDar



```
GNSS CRC Received: C339126B, Calculated: C339126B
GNSS Data Updated: N=0, E=0, H=108.389161914587
```

Figure 3. Figure of GNSS

5. Conclusion

This paper successfully designs and implements an integrated intelligent UGV control system unified within the MATLAB framework. Through a layered hardware architecture and multi-level communication protocol design, it resolves the challenge of integrating multi-source heterogeneous devices. By employing MATLAB/Simulink concurrently as both the algorithm development environment and the monitoring and communication hub, it overcomes the challenge of fragmented software frameworks. Field experiments validate the system's outstanding performance in perception, control, and human-machine interaction.

Future work will focus on: 1) incorporating visual sensors to achieve multimodal fusion perception for more complex environments; 2) integrating advanced SLAM and path planning algorithms within the existing framework to enhance full autonomy; 3) optimising wireless communication links and exploring 5G technology to support longer-range, lower-latency teleoperation. This research provides a replicable solution for rapidly constructing high-performance UGV development platforms.

References

[1] Cheung, C., Griffith, S., Wells, L., Gregory, D., Moore, M., Johannes, M., Hetherington, D., Walters, J., & Kang, S. (2023). Application of the Autonomous Ground Vehicle Reference Architecture to MBSE. In Proceedings of the Ground Vehicle Systems Engineering and Technology Symposium (GVSETS), NDIA, Novi, MI, Aug. 15–17, 2023. https://doi.org/10.4271/2024-01-4045

[2] Cai, Z., Liu, J., Chi, W., & Zhang, B. (2023). A Low-Cost and Robust Multi-Sensor Data Fusion Scheme for Heterogeneous Multi-Robot Cooperative Positioning in Indoor Environments. Remote Sensing, 15(23), 5584. https://doi.org/10.3390/rs15235584

[3] Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D., & Han, S. (2022). BEVFusion: Multi-Task Multi-Sensor Fusion with Unified Bird's-Eye View Representation. arXiv preprint arXiv:2205.13542. https://doi.org/10.48550/arXiv.2205.13542

[4] Qin, Y., Wang, C., Kang, Z., Ma, N., Li, Z., & Zhang, R. (2023). SupFusion: Supervised LiDAR-Camera Fusion for 3D Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2023). https://doi.org/10.48550/arXiv.2309.07084

[5] Pang, S., Morris, D., & Radha, H. (2020). CLOCs: Camera-LiDAR Object Candidates Fusion for 3D Object Detection. arXiv preprint arXiv:2009.00784. https://arxiv.org/abs/2009.00784

[6] SICK. (2025). LMS151-10100 | LMS1xx 2D LiDAR (Data sheet, Part No. 1047607). Retrieved from https://www.sick.com/media/pdf/0/40/840/dataSheet_LMS151-10100_1047607_zh.pdf

[7] Jin, M., Li, J., & Chen, T. (2024). Method for the Trajectory Tracking Control of Unmanned Ground Vehicles Based on Chaotic Particle Swarm Optimization and Model Predictive Control. Symmetry, 16(6), 708. https://doi.org/10.3390/sym16060708

[8] Martelli, S., Martini, V., Mocera, F., & Somà, A. (2025). Co-Simulation Model of an Autonomous Driving Rover for Agricultural Applications. Robotics, 14(9), 120. https://doi.org/10.3390/robotics14090120