International Core Journal of Engineering Volume 12 Issue 4, 2026
ISSN: 2414-1895 DOI: 10.6919/ICJE.202604_12(4).0007

Design and Implementation of a YOLOv5-Based Drone Campus Object Recognition System

Pengyu Li

School of Emergency Management and Safety Engineering, North China University of Science and Technology, Tangshan 063000, China

Abstract

Addressing the issues of low efficiency and limited coverage in traditional campus security patrols, as well as the fact that drone patrols can only collect data and lack real-time target recognition capabilities, this paper designs and implements a drone-based campus target recognition system based on the original YOLOv5 model.Developed using a cross-technology stack architecture comprising Vue, IDEA, and PyCharm, the system is divided into three core modules: image recognition, video recognition, and real-time video stream recognition. Without requiring algorithmic modifications or retraining of the YOLOv5 model, it achieves efficient recognition of campus scene data collected by drones through engineering integration techniques.Test results demonstrate that the system can reliably identify typical campus targets such as pedestrians, vehicles, and buildings. The image recognition accuracy reaches 85.2%, the video stream processing frame rate is maintained at 15–20 FPS, and real-time recognition latency is controlled within 450 ms, meeting the practical application requirements for scenarios such as campus drone patrols and security monitoring.This system facilitates the rapid engineering implementation of mature object detection models, providing a lightweight, highly available solution for smart campus security infrastructure.

Keywords

YOLOv5; UAV; Campus Recognition; Video Stream Processing; Cross-Stack Integration.

1. Introduction

As the development of smart campuses continues to advance, the need for intelligent upgrades in campus security management and routine patrols has become increasingly urgent. Drones, with their high maneuverability, wide patrol coverage, ease of operation, and controllable costs, have gradually replaced traditional manual patrol methods and become an essential tool in campus management.However, most current campus drone patrols remain limited to the stage of image data collection, with subsequent analysis of the collected footage relying on manual review. This approach suffers from issues such as delayed target identification, high labor costs, and low patrol efficiency. Target recognition technology is the key to addressing these pain points, but improving complex algorithms requires significant R&D resources and time, making it difficult to adapt to the practical needs of lightweight campus management scenarios.

As a leading object detection model, YOLOv5 offers technical advantages such as fast detection speed, low deployment barriers, and strong generalization capabilities. Its official base model can be directly applied to common object recognition scenarios, meeting basic identification needs without the need for additional data training. Based on this, this paper does not pursue breakthrough innovations at the algorithmic level but focuses on the engineering implementation and adaptation of mature models to campus scenarios, designing a dedicated system to support drone-based campus data recognition.By integrating the Vue front-end framework, a Java backend developed with IDEA,

ISSN: 2414-1895 DOI: 10.6919/ICJE.202604_12(4).0007

and a YOLOv5 inference module deployed via PyCharm, the system achieves comprehensive object recognition across all types of data-including static images, recorded videos, and real-time video streams-captured by drones. This provides immediate recognition support for campus drone inspections while offering practical engineering references for the scenario-specific application of similar mature models.

2. Issues with Traditional Campus Security Patrols

2.1 High Labor Costs and Inefficient Allocation of Patrol Resources

Traditional campus security patrols primarily rely on security personnel conducting foot or non-motorized patrols. To ensure full coverage of the campus, a large number of security personnel are often required to maintain 24-hour shift coverage, resulting in continuously rising labor costs.At the same time, there are clear limits to the effective coverage and efficiency of manual patrols. A significant amount of manpower is consumed in routine, repetitive patrols where no incidents occur, making it impossible to precisely allocate resources to high-risk scenarios such as campus perimeter boundaries, remote areas, and large-scale event sites;Long-term repetitive patrol duties also lead to issues such as visual fatigue and diminished attention among security personnel, directly resulting in a decline in inspection quality. This ultimately constitutes a severe waste of human resources and prevents the optimal allocation of inspection resources66[1] .

2.2 Difficulty in Meeting Emergency Patrol Demands During Peak Hours and Unforeseen Scenarios

During special periods such as school commute rush hours, large-scale campus cultural and sports events, exam weeks, and holidays, the volume of people and vehicles on campus surges within a short period. This significantly increases the likelihood of various abnormal incidents, including unauthorized vehicles entering the campus, crowds gathering, traffic congestion, and safety hazards.Traditional manual patrols have slow response times and cannot conduct comprehensive, large-scale inspections within a short timeframe. In most cases, they can only address issues after the fact, making it difficult to provide early warnings for abnormal incidents or intervene in a timely manner.Additionally, for emergencies such as scaling campus walls, sudden incidents in remote areas, and fire hazards, manual patrols suffer from delays in reaching the scene and an inability to transmit live footage immediately, making it difficult to meet the real-time requirements of campus security emergency response[2].[1]

2.3 Inability to Achieve Real-Time Monitoring and Object Recognition Around the Clock with No Blind Spots

Traditional manual patrols follow fixed routes and schedules, making it impossible to achieve 24/7 continuous, uninterrupted patrols across the entire campus. Areas such as rooftops, perimeter walls, secluded wooded areas, and underground parking garages are highly prone to becoming blind spots, creating potential safety hazards on campus[3] .At the same time, manual patrols rely solely on visual judgment, making it impossible to accurately and continuously identify and record objects in the footage. This leads to frequent missed or misjudged incidents involving illegal parking, fire hazards, unauthorized entry, and unusual gatherings of people;Furthermore, manual patrols cannot retain complete, traceable video data, making it difficult to conduct a full-process retrospective investigation after a security incident occurs and leaving a lack of comprehensive evidence to support incident resolution.

2.4 Lack of Systematic Analysis of Inspection Data and Support for Management Decisions

Traditional manual patrol records primarily rely on paper logs and verbal reports. Data collected during patrols is fragmented and lacks standardized formats, making it impossible to systematically aggregate, retain, or analyze it in depth. Campus security managers cannot use patrol data to accurately identify core patterns-such as peak times, high-risk areas, and the distribution of incident types-making it difficult to optimize patrol routes, adjust staffing levels, and improve security control

ISSN: 2414-1895 DOI: 10.6919/ICJE.202604_12(4).0007

measures in a targeted manner.Furthermore, the absence of quantifiable inspection data prevents standardized assessment and evaluation of security personnel's performance, ultimately forcing campus security management upgrades to rely solely on experience-based decision-making, lacking precise and effective data support[4].

In summary, the traditional campus security patrol model suffers from a series of issues, including inefficient allocation of patrol resources, insufficient emergency response capabilities, blind spots in comprehensive real-time monitoring, and a lack of systematic data analysis support.To address these issues, this paper proposes a YOLOv5-based drone campus target recognition system. By utilizing image capture devices mounted on drones and integrating three core modules-image recognition, video recognition, and real-time video stream recognition-the system achieves intelligent, comprehensive, and real-time control of campus patrols, providing a lightweight, highly available solution for smart campus security infrastructure.

3. System Requirements and Overall Design

3.1 Core Requirements Analysis

Considering the application scenarios of drones on campus, the system must meet three core requirements: First, compatibility-it must support JPG/PNG images, MP4/AVI videos, and real-time video streams via the RTSP protocol captured by drones; Second, real-time performance-real-time video stream recognition latency must be controlled within 500 ms to meet the immediate decision-making needs of drone patrols;third, ease of use, with a simple and intuitive front-end interface that supports visualization of recognition results and aligns with the operational habits of campus management personnel[5] . Additionally, the system must be developed based on the original YOLOv5 model to avoid R&D costs associated with algorithm modifications and ensure lightweight deployment.

3.2 Overall Architecture Design

The system adopts a separate front-end and back-end architecture. It is divided into three layers: the front-end interaction layer, the back-end service layer, and the model inference layer. Each layer realizes data interaction and collaborative work through standardized interfaces, which completely covers the entire process of UAV data reception, parsing, recognition, storage, and visual display.

The front-end interaction layer is developed using the Vue 3.0 framework and leverages the Element UI component library to build a visual operation interface. Its core functions include uploading drone-captured data (images/videos), receiving and playing real-time video streams, and visually rendering recognition results. It also provides user interaction capabilities, historical recognition data queries, and recognition log export functions.The backend service layer is built using the IDEA development environment with Java and the Spring Boot 2.7 framework. It handles core tasks such as request distribution, data format parsing, cross-module scheduling, storage of recognition results, and exception handling. By establishing standardized RESTful interfaces, it enables efficient communication with the frontend interaction layer and the model inference layer.The model inference layer is based on the PyCharm development environment, utilizing a Python 3.8 + PyTorch 1.12 runtime environment. It directly invokes the official YOLOv5 model, receives drone data transmitted from the backend, and performs object detection inference. Finally, it feeds back the recognition results-including object categories, bounding box coordinates, and confidence scores-to the backend service layer.

3.3 System Functionality Design

The Campus Object Recognition System supports login as either a user or an administrator. Administrators can configure user permissions, while all other functionalities remain consistent. The primary functions for both users and administrators include image recognition, video recognition, and real-time video stream recognition.see **Fig. 1**.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202604_12(4).0007

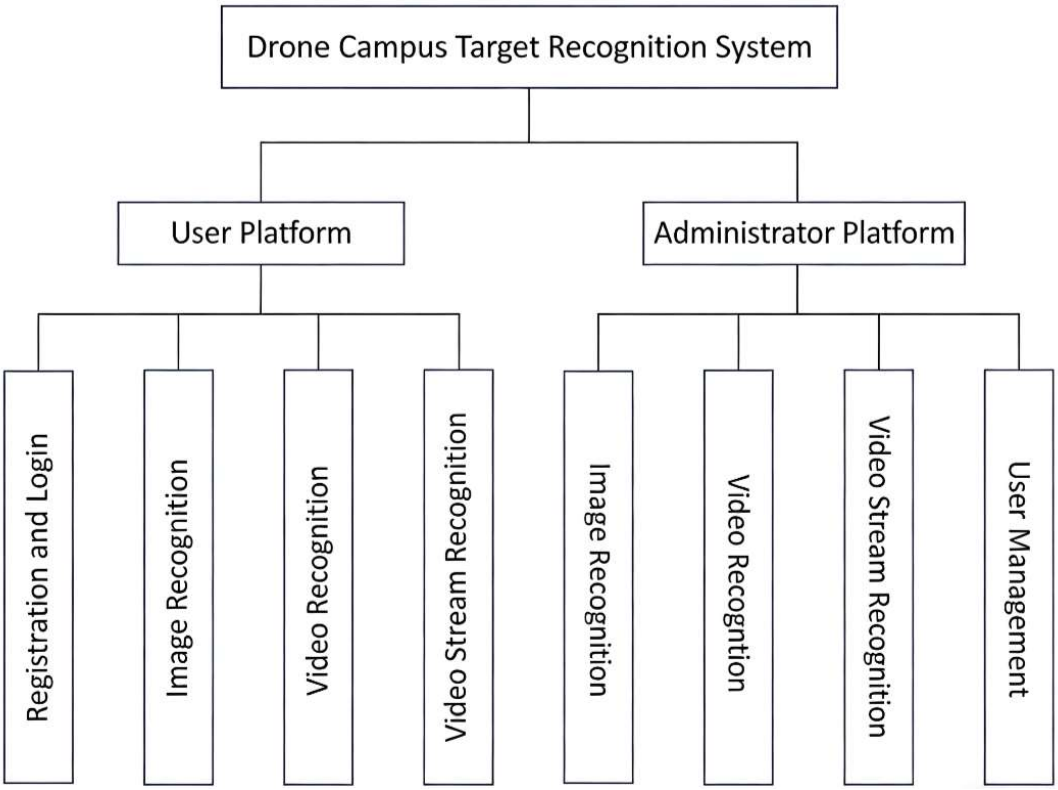

Fig. 1 System Functional Design Diagram

4. Design and Implementation of Core System Modules

Based on the practical requirements of campus drone recognition, the system is divided into three core functional modules. Each module independently performs specific functions and coordinates with others through the backend service layer to ensure comprehensive compatibility and adaptation to the various types of data collected by drones. The workflow of each core module is illustrated using flowcharts to clearly present the data processing logic.

4.1 Image Recognition Module

This module primarily processes static images of campus scenes captured by drones, enabling batch uploads, rapid recognition, and result annotation. The frontend uses Element UI components to build the image upload interface, supporting drag-and-drop uploads of single or multiple images. File formats are restricted to JPG or PNG, with a maximum size of 20MB, and a preview function is provided. Upon completion of the upload, the frontend uses Axios requests to transmit the image files to the backend interface in MultipartFile format.

Upon receiving the uploaded image files, the backend service layer first validates file validity (format, size). It then uses a format conversion tool to uniformly process the images into the RGB color space format. Subsequently, it invokes the model inference layer task via inter-process communication, passing the standardized image data to the YOLOv5 base model.After the model completes inference, it returns recognition results containing object categories, bounding box coordinates, and confidence thresholds. The backend service layer parses and formats these results, then uses OpenCV to draw object bounding boxes and confidence information on the original image, generating annotated result image files.Simultaneously, the backend stores key recognition information (upload time, number of objects, recognition accuracy, and image storage path) in a MySQL 8.0 database to ensure data persistence. Finally, the backend returns the annotated images and structured recognition results to the frontend via an asynchronous interface. The frontend uses the Canvas component to render the annotation boxes and object information, providing a visual representation of the recognition results,see **Fig. 2**.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202604_12(4).0007

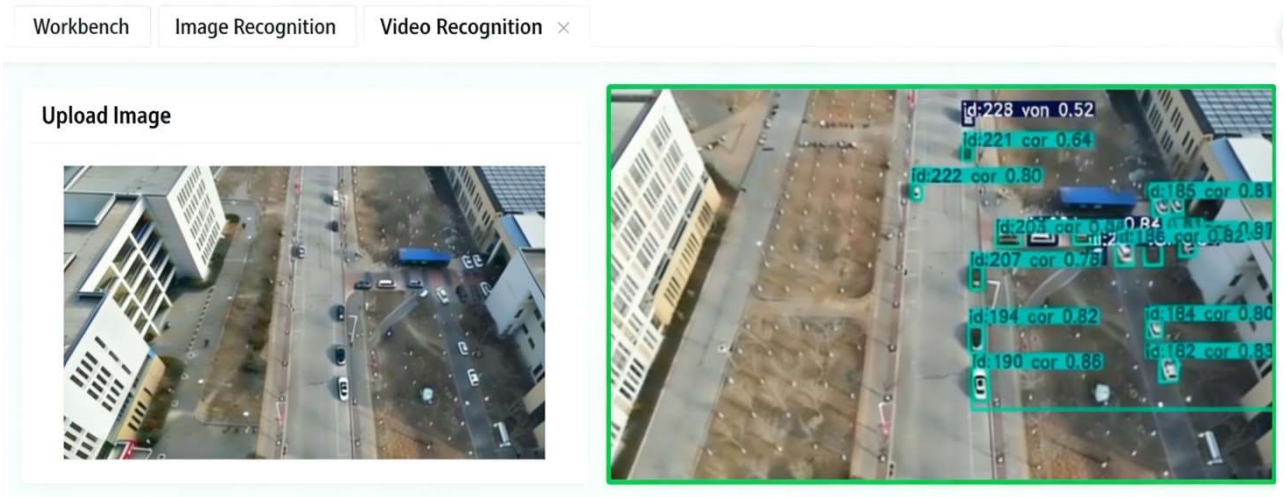

Fig. 2 Image Recognition Interface

4.2 Video Recognition Module

For campus inspection videos recorded by drones, this module implements video parsing, frame extraction, frame-by-frame recognition, and result synthesis. The frontend supports video uploads in MP4/AVI formats, providing operations such as video progress preview, pause, and variable-speed playback, while also allowing users to set the recognition accuracy (confidence threshold adjustable between 0.5 and 0.9).Upon receiving video files, the backend decodes the video using the FFmpeg tool, extracts video frames at a fixed frame rate (default 25 FPS), and stores the frame images in a temporary folder to avoid excessive memory consumption[6] .

To improve video frame processing efficiency, this module employs a multi-threaded parallel processing mechanism. It uses a thread pool to schedule multiple model inference processes, performing batch parallel recognition on the extracted video frames , effectively reducing the overall processing time[7] .After the original YOLOv5 model completes object recognition for each video frame, it feeds the results back to the backend service layer. The backend uses OpenCV to fuse the recognition results with the original video frames, accurately annotating object information and confidence scores. Once fusion is complete, FFmpeg is used to re-encode the processed frames into a video file, maintaining the original video's resolution, frame rate, and other parameters to ensure smooth playback.Once video processing is complete, the backend service layer stores the processed video files and detailed recognition logs (including per-frame recognition results and processing time). The frontend provides users with online playback, download, and frame-by-frame viewing capabilities for the processed video, supporting secondary verification of recognition results,see **Fig. 3**.

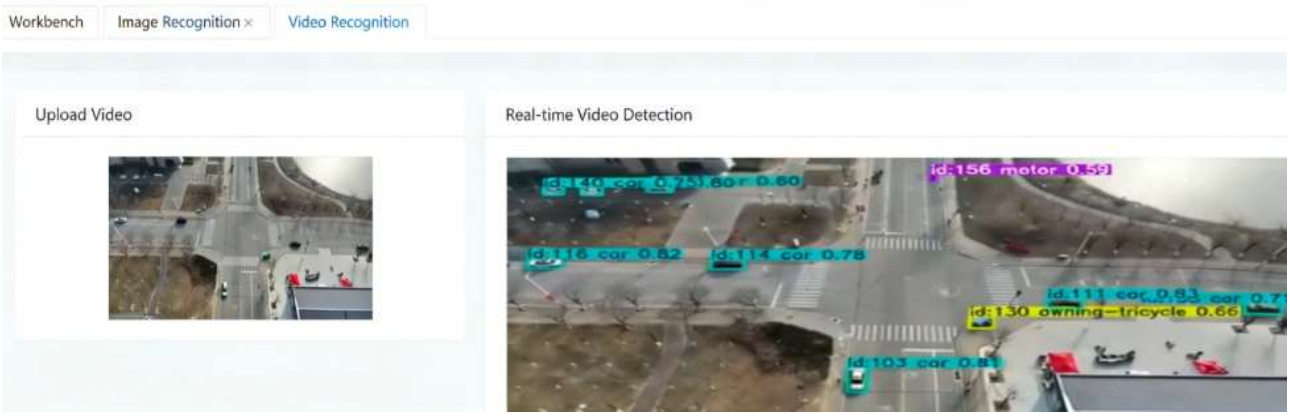

Fig. 3 Video Recognition Interface

ISSN: 2414-1895 DOI: 10.6919/ICJE.202604_12(4).0007

4.3 Real-time Video Stream Recognition Module

This module is a core highlight of the system. It adapts to drone RTSP real-time video transmission streams to achieve low-latency recognition and real-time display, meeting the immediacy requirements of campus patrols.The frontend uses the Video.js component to connect to the drone's RTSP stream, configuring the transmission protocol and decoding method to ensure smooth playback. The backend uses the JavaCV toolkit to monitor the drone's video transmission port, receive real-time stream data, perform demultiplexing, extract video frames, and convert them into a format recognizable by the model.

To address the low-latency requirements of real-time video stream recognition, this module employs an inter-frame sampling strategy (default: 1 frame sampled every 3 frames, dynamically adjustable based on hardware performance). This effectively reduces the model's inference load while maintaining recognition accuracy, keeping end-to-end recognition latency within a reasonable range.The sampled frames are rapidly fed into the YOLOv5 base model for inference. The inference results are immediately fed back to the backend service layer, where they are quickly parsed to generate standardized annotation information. This annotation information is pushed in real time via the WebSocket protocol, ensuring that the frontend recognition results are rendered in sync with the live video feed.Additionally, this module features a built-in anomaly detection alert function. It is pre-configured with campus violation scenarios (such as unauthorized vehicles entering the campus, crowd gatherings, or people scaling fences). When a corresponding anomaly is detected and the confidence level exceeds the set threshold, the system automatically triggers audible and visual alarms. It also records information such as the alarm time, target location, and on-site video screenshots, providing timely early warning support for campus security personnel,see **Fig. 4**.

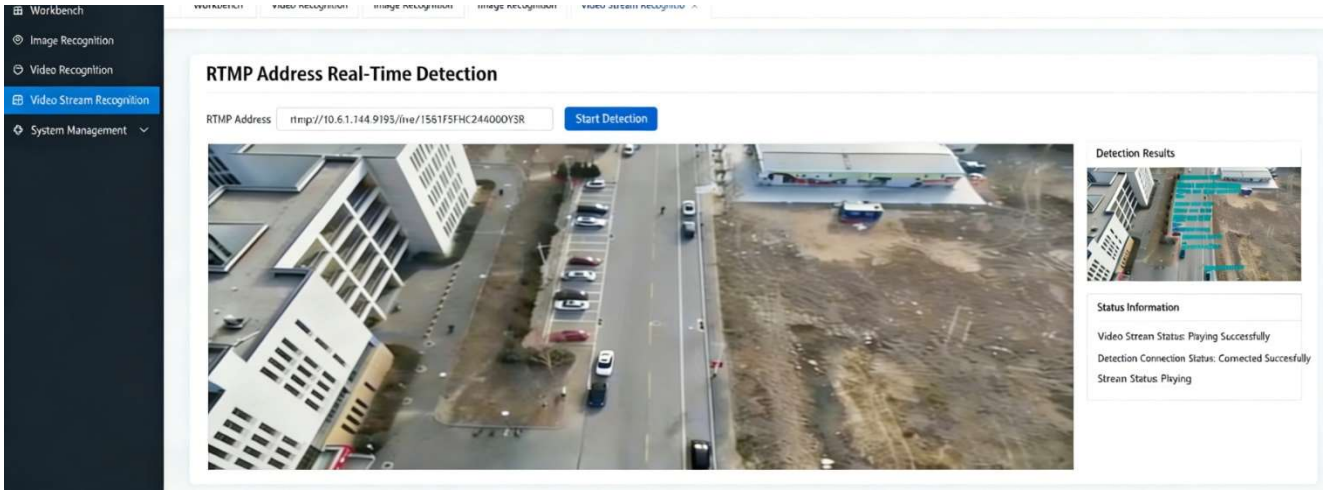

Fig. 4 Real-time Video Stream Recognition Interface

5. System Testing and Results Analysis

5.1 Test Environment

To verify system performance, a test environment was set up: In terms of hardware, the CPU is an Intel Core i5-13400F,the GPU is an NVIDIA RTX 4060 (8GB VRAM), and the system has 16GB of RAM; for software, the operating system is Windows 11, the frontend uses Vue 3.0, the backend uses Spring Boot 2.7, the database is MySQL 8.0, and the model inference environment consists of Python 3.8 and PyTorch 1.12; the drone model is the DJI M3E, with a video transmission resolution of 1080p.

5.2 Test Cases and Results

The test selected three typical campus scenarios (classroom building area, playground, and campus roads), collecting 500 drone images, 10 video clips (each 5–10 minutes long), and 1 hour of real-time

ISSN: 2414-1895 DOI: 10.6919/ICJE.202604_12(4).0007

video transmission. Functional and performance tests were conducted on the system's three major modules, with the following results:

(1)Functional Testing: The image recognition module accurately identified targets such as pedestrians, vehicles, and buildings, with no missed or misidentified instances; the video recognition module produced clear composite video frames with accurate annotation boxes and no stuttering; the real-time video streaming module maintained a stable latency of 300–450 ms, and the alarm function triggered promptly, meeting the requirements for drone inspection.

(2)Performance Testing: Under the aforementioned hardware configuration, the system performance metrics were as follows: Image recognition took an average of 0.8 seconds per image, with an 89.2% target recognition accuracy rate; during video recognition, the processing frame rate remained stable at 15–20 FPS, and a 10-minute campus inspection video took an average of 4.5 minutes to process;The end-to-end latency for real-time video stream recognition was 300–450 ms, with CPU utilization controlled at 40%–50% and GPU utilization at 60%–70%. Throughout the process, there were no program crashes, frame stuttering, or data loss, and the system demonstrated excellent operational stability.

5.3 Results Analysis

The test results clearly demonstrate that the drone-based campus target recognition system built on the original YOLOv5 model effectively meets the application requirements of drone campus patrol scenarios, with all three core modules functioning properly and performing stably.Although this system did not implement algorithmic improvements to the YOLOv5 model, engineering optimizations such as multi threaded parallel processing and frame-interval sampling significantly enhanced the system's real-time performance and processing efficiency, fully meeting the requirements for recognition accuracy and response speed in campus security patrols.A limitation of this system is that the original YOLOv5 model exhibits low recognition accuracy for small campus objects (such as stray cats, ground debris, and small signage). This is primarily because the model's training data did not include campus-specific scenarios. Future work could involve fine-tuning the model using a dedicated campus data set to further optimize recognition performance.

6. Conclusion

Guided by the needs of smart campus security patrols, this paper has developed a drone-based campus target recognition system based on the original YOLOv5 model through cross-technology stack integration. The system focuses on the engineering implementation and scenario adaptation of a mature model, avoiding the R&D costs associated with complex algorithm improvements. It is divided into three major modules: image recognition, video recognition, and real-time video stream recognition, and features comprehensive functionality, ease of operation, strong real-time performance, and lightweight deployment.Test results validate the system's feasibility and practicality in campus drone recognition scenarios. It effectively addresses the issues of delayed recognition and low efficiency in traditional drone patrols, providing technical support for smart campus security management. Additionally, it offers a practical reference for the scenario-specific engineering applications of similar mature object detection models.

Future optimizations can be pursued in two areas: first, fine-tuning the YOLOv5 model using campus-specific datasets to improve the recognition accuracy of small targets; second, expanding system functionality by integrating drone path planning with recognition results to enable a unified "recognition-alert-dispatch" workflow, thereby further enhancing the intelligence of campus management.

References

[1] C.L. Xu: Current Status and Application of Campus Security Operation and Maintenance System, China Security & Protection, Vol. (2018) No.3, p.43-47. (in Chinese)

ISSN: 2414-1895 DOI: 10.6919/ICJE.202604_12(4).0007

[2] S.H. Liao, J.D. Jiang and C.F. Yang: Integration of LoRa-enabled IoT infrastructure for advanced campus safety systems in Taiwan, Internet of Things, Vol. 28 (2024), p.101347.

[3] T. Chen: Application of Video Surveillance in Campus Security Protection System, Nonferrous Metallurgical Design and Research, Vol. 46 (2025) No.5, p.48-51. (in Chinese)

[4] B.X. Wang, H.J. Li, Z.Y. Zheng, et al.: Analysis and Control of Campus Safety Hazards Based on Risk Pre-control, Computer Knowledge and Technology, Vol. 17 (2021) No.35, p.285-287. (in Chinese)

[5] A.M. Gaber, R.A. Rashid, N. Nasir, et al.: Development of an autonomous iot-based drone for campus security, ELEKTRIKA-Journal of Electrical Engineering, Vol. 20 (2021) No.2-2, p.70-76.

[6] M.U. Yaseen, A. Anjum, M. Farid, et al.: Cloud-based video analytics using convolutional neural networks, Software: Practice and Experience, Vol. 49 (2019) No.4, p.565-583.

[7] X.X. Liu: Multi-threaded Parallel Loading of Big Data Files, Journal of Guangxi Normal University for Nationalities, Vol. 34 (2017) No.3, p.134-136. (in Chinese)