Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 17, No. 1, 2026

Design and Implementation of a Smart Scenic Area Tourist Behavior Safety Monitoring System based on HiSilic

Dan Yin [1], Siwen Sun [2], Ting Zhang [1], Jianli Gong [1]

[1] Wuhan University of Technology and Business, Wuhan, Hubei, China
[2] South-Central Minzu University, Wuhan, Hubei, China

Abstract: Inspired by the technical capability of the HiSilicon Hi3516DV300 camera to efficiently run the YOLOv3-tiny model on embedded devices, this solution offers significant advantages of high accuracy, low latency, and low power consumption, providing a valuable reference for the application of lightweight edge intelligence devices in scenic area monitoring. This study analyzes and designs a smart scenic area tourist behavior safety monitoring system based on an “edge-cloud” collaborative architecture. The hardware terminal adopts the HiSilicon Hi3516DV300 smart camera, equipped with an optimized YOLOv3-tiny lightweight object detection model accelerated by TensorRT, to achieve local real-time detection of human bodies and key targets within the scenic area. Simple tracking and trajectory analysis are combined to complete crowd flow statistics and density warning. The model is also tailored for head/body detection and deployed at the edge for real-time counting. On the cloud side, the Alibaba Cloud IoT platform provides device management, data storage, visual analytics, and remote interaction functionalities.

Keywords: HiSilicon; Tourist Behavior Monitoring; Crowd Flow Statistics; Edge-cloud Collaboration; Edge Computing.

1. Introduction

1.1. Research Background

Cultural tourism industry is gaining significant numbers of visitors, and monitoring tourist behavior safety and crowd density are imperative requirements of smart scenic area management [4]. Traditional safety monitoring relies on manual patrols and fixed cameras, which require slow response times, limited coverage and are unable to perform real-time crowd counting or dangerous behavior recognition, which makes them hard to meet the management requirements of large-scale tourism attractions [5].

Smart monitoring models for scenic areas are currently aimed at observing the environment and performing low-level passenger flow statistics. They do not accurately detect human target detection, trajectory analysis and real-time density detection. Also, the purely cloud based image recognition models are limited with network bandwidth and latency, and cannot monitor millisecond real-measurely. Moreover, ordinary embedded devices cannot compute deep learning models for complex object detection and behaviour analysis [9] [14].

The emergence of embedded AI chips such as the HiSilicon Hi3516DV300 provides the hardware foundation for deploying lightweight deep learning models at the edge [13]. The YOLO [1] series of lightweight detection models, after optimization, can achieve a good balance between accuracy and speed. The cloud-edge collaborative architecture enables rational distribution of computational tasks between edge devices and the cloud platform, balancing monitoring real-time performance with in-depth data analysis, thus becoming an important technical direction for smart scenic area tourist behavior safety monitoring [6].

1.2. Research Significance

This research designs a smart scenic area tourist behavior safety monitoring system based on the HiSilicon Hi3516DV300, integrating embedded AI, the Internet of Things, computer vision, and cloud-edge collaboration technologies [3]. The system achieves integrated intelligent monitoring including real-time detection of human bodies/heads, target tracking, trajectory analysis, crowd flow statistics, and density warning within the scenic area. [4]

The system performs low-latency object detection and preliminary data analysis at the edge, while the cloud side enables multi-device management, data integration, and visual presentation [10]. This establishes a scenic area safety management loop of “real-time perception → intelligent analysis → warning response → remote control.” Such a system upgrades tourist safety monitoring from “passive surveillance” to “active warning,” improves the efficiency of crowd control and safety incident handling, and provides a standardized architectural reference for designing and implementing edge intelligence systems in smart scenic areas. It thus carries significant theoretical value and engineering practical significance.

2. Related Work

Domestically, supported by smart scenic area construction policies, IoT and big data technologies have been applied to passenger flow statistics and environmental monitoring. Liang Yi designed an MQTT-based scenic area ecological monitoring module to address data accuracy and visualization issues. Peng Xianhua built a LoRa environmental monitoring system for data collection and reporting. Fan Luqiao and others implemented scenic area data visualization based on Python+Echarts, providing ideas for backend processing and display of scenic area monitoring. However, existing systems mostly lack real-time visual detection and behavior analysis of tourist targets.

Internationally, smart scenic area development started earlier, with researchers integrating IoT and AI technologies into scenic area management. Zhao J et al. designed a remote air environment monitoring system [11]. Pu Z et al. utilized WiFi to obtain passenger flow data [12]. Some studies have attempted to apply computer vision technology to crowd flow statistics in scenic areas, but the deployment of lightweight models at the edge for low-latency human detection and trajectory analysis still requires improvement. [8]

Although existing research provides ideas for sensor collaboration, communication protocols, and data visualization for scenic area monitoring, there are deficiencies in real-time visual detection at the edge, accurate head/body recognition, and dynamic crowd density warning [2]. Based on existing research, this project constructs an "edge-cloud" full-link framework, takes the HiSilicon Hi3516DV300 as the edge core, optimizes the YOLO [1] [7] model for real-time target detection in scenic areas, and combines the Alibaba Cloud platform for data visualization and remote control, filling the technical gap of edge intelligent vision monitoring in smart scenic areas.

3. System Design

3.1. Overall Architecture

The system adopts an edge-cloud collaborative layered architecture consisting of three tiers: "edge layer – communication layer – cloud layer." This design achieves an organic combination of edge-based real-time perception and cloud-based intelligent analysis and control of tourist behavior in scenic areas. It breaks data silos and builds a complete closed loop of "acquisition → detection → analysis → warning → control," meeting the core requirements of scenic area safety management: "real-time monitoring, intelligent decision-making, rapid response." The architecture is illustrated in Figure 1.

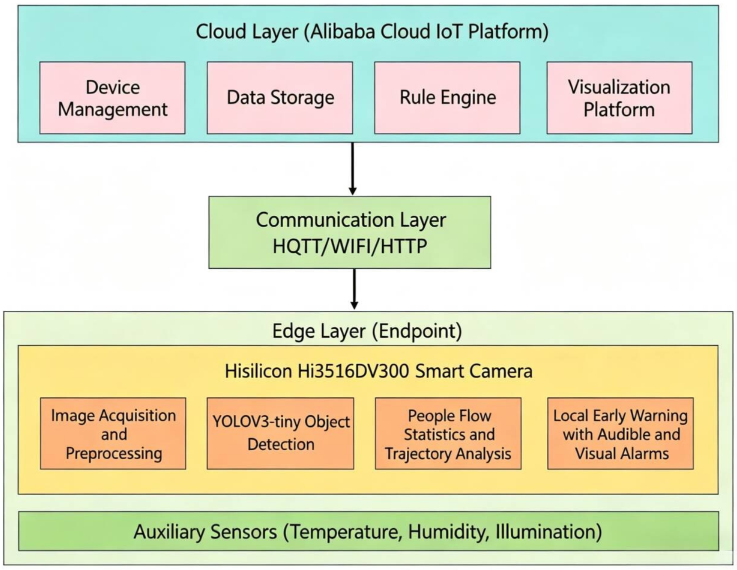

Figure 1. Overall Architecture Design

3.1.1. Edge Layer

The main device is a HiSilicon Hi3516DV300 smart camera running a TensorRT accelerated and optimized YOLOv3–tiny [7]. It is responsible for video frames, real-time heads and bodies detection, basic target tracking and trajectory analysis, edge crowd flow statistics, density calculation, and local warning, all done in millisecond-latency. Auxiliary sensors (temperature, humidity, light) are also employed to collect environmental information and provide multi-source information for monitoring.

3.1.2. Communication Layer

MQTT and WiFi protocols are used for two-way asynchronous communication between edge devices and cloud.The two protocols reset after reconnection and offline caching, and the transmission is reliable even in the challenging networks of mountains. Additionally, HTTP/HTTPS protocols are adopted for device authentication and file transfers.

3.1.3. Cloud Layer

The cloud layer runs on Alibaba Cloud IoT and provides device access management, detection results and video, rule engine analysis and visualization services from ECharts. It supports cooperation of multiple edge devices and data interfaces for mobile apps to monitor and control a remote device.

3.2. Three-Tier System Architecture

We built the smart scenic area tourist behavior safety monitoring system with a three-level system consisting of "hardware terminal – cloud platform – software terminal" and "smart scenic area – tourist behavior – safety – terminal's" core hardware terminal.

3.2.1. Hardware Terminal

The hardware terminal implements device identity authentication, data reception and storage, rule engine configuration (e.g., density warning thresholds) and remote model parameter update. The software terminal includes image acquisition module, object detection computing module, WiFi communication module, and auxiliary environmental sensors. It records image capture, object recognition, crowd flow statistics, local warning and data upload, receives control commands from the cloud, and performs the actions.

3.2.2. Cloud Platform

The Alibaba cloud platform implements device ID authentication, information receive and store, rule engines (e.g., density alert thresholds) as well as remote model parameters updates. It provides stable data support and service interfaces for the software terminal.

3.2.3. Software Terminal

The software terminal has a web visualization system built with ECharts, and an Android mobile app. The web visualization software system provides the park manager with real-time views of object detection results, crowd density heatmaps, trajectory charts, and warning information. The mobile app supports remote viewing of the monitoring information, warning notifications and device control commands which allow the park management and control.

3.3. Functional Architecture

Based on the core tourism behavior safety tracking, the system has six functional modules. Each module is responsible for its own tasks and clear interfaces, supporting independent development and deployment.

Images Acquisition and Preprocessing module: captured video frames from the region, preprocesses the imagery (white balance, noise reduction, size normalization) with HiSilicon ISP unit, adapts to input requirements of detection model, and improves detection robustness in challenging scenarios.

Object Detection and Optimization Module: Implement our basic target tracking and trajectory analysis with the help of YOLOv3-tiny model [7] [11] (e:g., high precision detection of heads and bodies and outputting target category, position, and confidence information.

Tracking Analysis and Crowd Flow Statistics Module: (e-g., real-timing the crowd counting and regional crowd density estimation through cross-frame target matching).

Warning and Local Control Module: Signals for crowd density and dangerous behavior; local audible and visual warnings when the thresholds are exceeded; copy warnings to the cloud.

Edge-Cloud Communication Module: Connectes bidirectional data from edge to cloud; upload detection data, crowd flow data, warning information; upload cloud control commands and model parameters.

Visualization and Remote Control Module: Web visualization and a mobile app for intuitive display of monitoring data, alerting and remote device control and model parameter settings.

4. Database Design

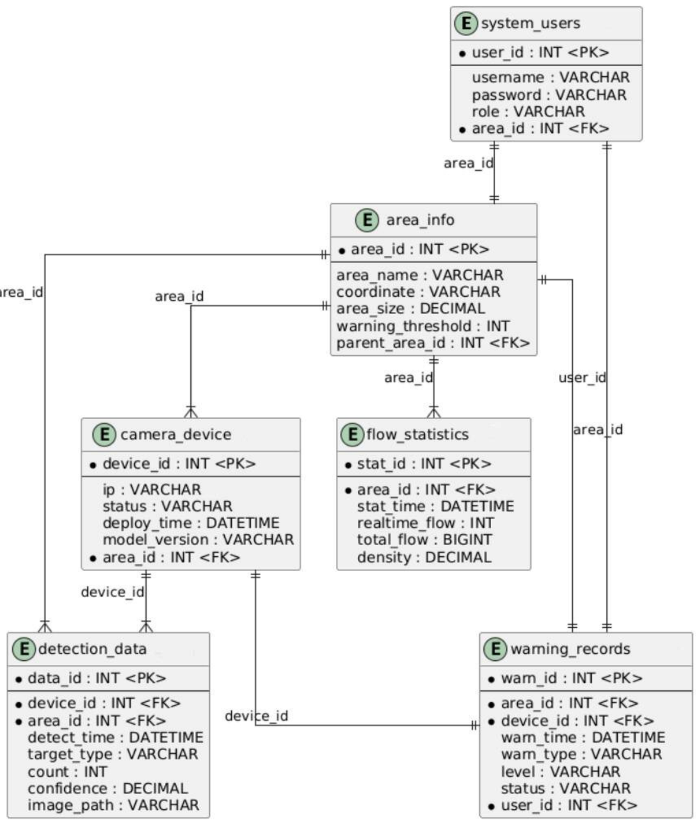

Figure 2. The design of the data tables

The system database architecture comprises six core data tables linked via foreign keys. This design forms a closed loop for data collection, analysis, warning, and user management in tourist behavior monitoring for scenic areas, supporting the intelligent operation of the system. The table structures and relationships are described below:

area_info table: Defines different monitoring areas in the scenic area. Uses `parent_area_id` to build area hierarchy (e.g., scenic area → attraction → monitoring point). Stores area ID, area name, location coordinates, area size, density warning threshold, etc.

camera_device table: Manages HiSilicon Hi3516DV300 camera devices. Stores device ID, associated area ID, device IP, status, deployment time, model version, etc. Links to `area_info` via `area_id`.

detection_data table: Stores object detection data uploaded from the edge. Contains data ID, device ID, area ID, detection time, target type (head/body), target count, confidence, detection frame screenshot URL, etc. Links to `camera_device` and `area_info`.

flow_statistics table: Stores crowd flow data for each area in the scenic area. Contains statistics ID, area ID, statistics time, real-time passenger flow, cumulative passenger flow, area crowd density, etc. Links to `area_info` to provide data for density warning.

warning_records table: Records warning information in the scenic area. Contains warning ID, area ID, device ID, warning time, warning type (density exceedance / dangerous behavior), warning level, processing status, processing user ID, etc. Links to `area_info`, `camera_device`, and `system_users`.

system_users table: Manages system users. Stores user ID, username, password, permission level, responsible area ID, etc. Links to `area_info` via `area_id` to enable fine-grained user permission management.

The design of the data tables is shown in Figure 2.

The data tables are linked through foreign keys, enabling relational management of monitoring areas, devices, detection data, crowd flow data, warnings, and users. This supports fast data querying, statistics, and traceability, and provides data support for scenic area management decision-making.

5. Visualization System Design

5.1. Overall Layout of the Visualization System

In designing the visualization system, HTML, JS, and CSS are used to construct the overall page layout. This system name is provided by Smart Scenic Area Monitoring System and information is displayed by <h2 id=yanwuCount <h3>, <h 2 id=wenduCount < h2> etc. ECharts library is used: line charts (series: {type: line}) show the temperature and humidity, and pie charts (Series: { type: pie}) display alarm types. We provide the code for building the overall layout and show the user interface in Fig 3.

```
<div class="b_left_box">
    <img class="t_l_line" src="img/left_line.png" alt="">
    <div id="chart_2" class="echart" style="width: 100%; height: 3.6rem;"></div>
    <img class="t_r_line" src="img/right_line.png" alt="">
</div>
```

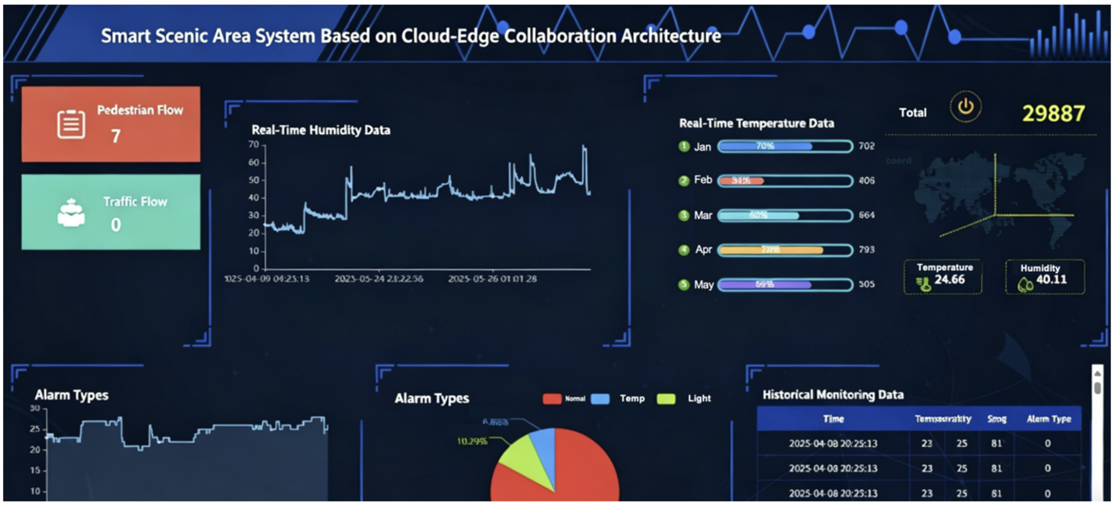

Figure 3. Visual System Layout Diagram

5.2. Traffic Statistics

Two methods *gethebeiNumber1* and *getshebeihmNumber2* will query the latest pedestrian flow and vehicle flow information from the *LOGS* table of the database. *getshebeeiNumber1* will query SQL query by taking the *ren_flow* column and ordering *currytime* descending in order to get the latest passenger flow records. *LIMIT1* will only return the latest driver flow records; *gethehmNumber2* will simply return the newest pedestrian flow records for the purposes of real time. The core code of real world driver and vehicle flows is shown below.

Core code:

```java
@Select("SELECT ren_flow as count FROM LOGS ORDER BY currytime DESC LIMIT 1")
List<WarnVo> getshebeiNumber1();

@Select("SELECT car_flow as count FROM LOGS ORDER BY currytime DESC LIMIT 1")
List<WarnVo> getshebeiNumber2();
```

Figure 4. Core code

6. App Design

6.1. Overall Layout of the App

Android development was used to create display components for temperature, humidity, light, pedestrian flow, vehicle flow, alarm type, user input data, and control methods. A *LinearLayout* is used to set a horizontally centered container. On the left, a *TextView* displays the label (e.g., “Temperature/Humidity”) with a specified font, while the right side shows data values collected from the STM32. Spacing is controlled via *android:layout_margin*, and operation interfaces are provided via *android:id*. A key code snippet is shown below, and the resulting interface is shown in Figure 5.

```xml
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:orientation="horizontal">
    <TextView
        android:id="@+id/LightStatus2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="16dp"
        android:layout_marginEnd="3dp"
        android:text="Humidity:"
        android:textSize="20sp"
        android:textStyle="bold" />
```

Figure 5. Overall layout diagram of the APP

6.2. Real-time Data Display

```java
handler = new Handler() {
    public void handleMessage(Message msg) {
        super.handleMessage(msg);
        switch (msg.what) {
            case 1: break;
            case 2: break;
            case 3:
                UTF8Buffer msg = new UTF8Buffer(object.toString());
                String message = msg.obj.toString();
                Log.d("nicecode", "handleMessage: " + message);
                try {
                    JSONObject jsonObject = new JSONObject(message);
                    String temp = jsonObject.optString("temp");
                    String humi = jsonObject.optString("humi");
                    String light = jsonObject.optString("light");
                    String ren = jsonObject.optString("ren_flow");
                    String car = jsonObject.optString("car_flow");
                    int warning = jsonObject.optInt("warning_flag");
                    // ... further processing
                } catch (JSONException e) {
                    e.printStackTrace();
                }
                break;
        }
    }
```

Figure 6. The key code snippet

When the message type *msg.what* equals 3, MQTT data is processed. The message content is converted to JSON format, and data such as temperature, humidity, light, pedestrian flow, and vehicle flow are parsed. The key code snippet is shown in Fig 6.

Acknowledgments

This work was supported by the National College Student Innovation and Entrepreneurship Training Program [Grant No. 202413242003] of Wuhan Technology and Business University.

References

[1] Ren, K., et al. (2020). A systematic study of tiny YOLO3 inference: Toward compact brainware processor with less memory and logic gate. IEEE Access, 8, 142931–142955. https://doi.org/10.1109/ACCESS.2020.2972284.

[2] Adewumi, A., & Saraswat, D. (2025). Assessing TensorRT optimization for real-time object detection in agriculture. In 2025 ASABE Annual International Meeting (Paper No. 2500469). ASABE. https://doi.org/10. 13031/aim.2025. 2500 469.

[3] Qian, B., Xuan, Y., Wu, D., et al. (2025). Edge-cloud collaborative streaming video analytics with multi-agent deep reinforcement learning. IEEE Network, 39(1). https://doi. org/ 10. 1109/MNET.2025.3492761.

[4] Dong, Q. (2024). Safety monitoring system for tourist scenic spots based on crowd scene type recognition. International Journal of Security and Networks, 19(3), 128–137. https://doi.org/10.1504/IJSN.2024.139652.

[5] Huang, S., Ji, J., Wang, Y., Li, W., & Zheng, Y. (2023). A machine vision-based method for crowd density estimation and evacuation simulation. *Safety Science*, *167*, 106285. https:// doi. org/10.1016/j.ssci.2023.106285.

[6] Tariq, M. A., Ajmal, M., Jo, E., et al. (2024). Revolutionizing surveillance: A brief survey of edge AI terminals. In 2024 6th International Conference on Artificial Intelligence in Information and Communication (ICAIIC) (pp. 281–285). IEEE. https://doi.org/10.1109/ICAIIC60668.2024.10446789.

[7] Calì, R., Falaschetti, L., & Biagetti, G. (2025). Optimized implementation of YOLOv3-Tiny for real-time image and video recognition on FPGA. Electronics, 14(20), 3993. https://doi.org/10.3390/electronics14203993.

[8] Tariq, M. A., Ajmal, M., Jo, E., et al. (2024). Revolutionizing surveillance: A brief survey of edge AI terminals. In 2024 6th International Conference on Artificial Intelligence in Information and Communication (ICAIIC) (pp. 281–285). IEEE. https://doi.org/10.1109/ICAIIC60668.2024.10446789.

[9] Lema, D. G., Usamentiaga, R., & García, D. F. (2024). Quantitative comparison and performance evaluation of deep learning-based object detection models on edge computing devices. *Integration*, *95*, 102127. https://doi.org/ 10. 1016/j. integration.2024.102127.

[10] Martin-Salinas, I., Badia, J. M., Valls, O., et al. (2024). Evaluating and accelerating vision transformers on GPU-based embedded edge AI systems. *The Journal of Supercomputing*, *80*, 28461–28494. https://doi.org/10.1007/s11227-024-05987-9.

[11] Wijayanti, K., Mutiara, G. A., Suryawardani, B., Ervina, E., & Kusuma, G. P. (2025). Non-intrusive real-time tourist crowd monitoring for overtourism mitigation using YOLOv8-based head detection and tracking. *Journal of Robotics and Control*, *6*(4). https://doi.org/10.18196/jrc.v6i4.26396.

[12] Dionis-Ros, A., Vila-Francés, J., Magdalena-Benedito, R., Mateo, F., & Serrano-López, A. J. (2024). Multimodal video analysis for crowd anomaly detection using open access tourism cameras. *Applied Sciences*, *14*(23), 11075. https://doi. org/ 10.3390/app142311075.

[13] Veesam, S. B., Rao, B. T., Begum, Z., et al. (2025). Multi-camera spatiotemporal deep learning framework for real-time abnormal behavior detection in dense urban environments. *Scientific Reports*, *15*, 26813. https://doi.org/10.1038/s41598-025-12388-7.

[14] Li, J., & Fu, A. (2025). Anomaly behavior detection based on deep learning in an IoT environment. *Sensors*, *25*(24), 7605. https://doi.org/10.3390/s25247605.