UTHM Penerbit
Universiti Tun Hussein Onn Malaysia

JOURNAL OF SCIENCE AND TECHNOLOGY
ISSN: 2229-8460 e-ISSN: 2600-7924

Vol. 18 No. 1 (2026) 187-201
https://publisher.uthm.edu.my/ojs/index.php/jst

An IoT Platform for the Management of Fruit Fly Electric Traps

Vy-Khang Tran[1], Quoc-Duy Tran[1], Chi-Ngon Nguyen[2], Van-Khanh Nguyen[1,2]*

[1] *PLC Technology and IIoT Laboratory, College of Engineering, Can Tho University, Can Tho City, 94000, VIETNAM*
[2] *College of Engineering, Can Tho University, Can Tho City, 94000, VIETNAM*

*Corresponding Author: vankhanh@ctu.edu.vn
DOI: https://doi.org/10.30880/jst.2026.18.01.015

Article Info

Received: 11 January 2026
Accepted: 2 May 2026
Available online: 27 June 2026

Keywords

Internet of Things (IoT), web-based platform, remote fruit fly monitoring, real-time data collection, oriental fruit fly

Abstract

The Oriental fruit fly is a major agricultural pest that directly impacts crop yield and quality. However, current monitoring methods are still largely manual, lacking automation and continuity. This study proposes a remote monitoring platform that integrates automated electronic traps, Long Range (LoRa) communication, and an intuitive web interface. The system comprises sensor stations that transmit real-time data to a customized remote server via the internet, where the data are processed, stored using a JSON file-based NoSQL structure, and visualized through a web-based dashboard. The server supports concurrent monitoring of multiple stations, historical data access, and real-time device status tracking. Secure data transmission is ensured through Hypertext Transfer Protocol Secure (HTTPS) / Transport Layer Security (TLS) protocols integrated with Cloudflare services. Field experiments at two test stations achieved detection accuracies of 86.76% and 93.50%, respectively. The platform demonstrates strong potential for practical deployment in automated and large-scale pest management. Future enhancements will focus on integrating early warning capabilities, advanced data analytics, and scalable architecture to better support smart agriculture applications.

1. Introduction

The Oriental fruit fly poses a critical and immediate threat to global agricultural production, severely impacting vegetables and fruit trees. With alarmingly high population density and species diversity, these pests aggressively attack an extensive range of host plants. Annual yield losses for key export crops are devastating, ranging from 40% to 80% depending on location, crop type, and season [1–3]. Of all species, Bactrocera dorsalis (B. dorsalis) stands out as the most destructive fruit fly worldwide, its prevalence reaching crisis levels in Southeast Asia [4]. Its exceptional adaptability and powerful dispersal capacity allow B. dorsalis to invade new territories rapidly, frequently aided by the trade and transport of infested produce [5]. As a result, developing robust and urgent strategies for control and prevention of fruit fly dissemination is imperative.

Integrated Pest Management (IPM) strategies have been widely applied to control fruit flies [6–9]. Manual monitoring, using traditional traps with attractants, remains common but is labor-intensive, repetitive, and error-prone [10]. These methods lack continuous spatiotemporal monitoring, limiting timely interventions. To overcome this, many studies propose electronic traps that automatically detect, identify, and count fruit flies by integrating sensors and microcontrollers into conventional designs [10]. These systems use features like shape,

wingbeat frequency, and infrared reflection [11]. Combining such traps with IoT technology enables remote, automated monitoring, reducing fieldwork and enhancing efficiency in large-scale agriculture.

IoT has been widely adopted in agriculture, notably for tasks like irrigation and livestock monitoring [12]. A key application is remote automatic pest monitoring, which has gained research interest [13]. In fruit fly control, IoT-based systems often use electronic traps linked to a gateway that sends data to a server via wireless networks, a structure seen in several studies [14–16]. For example, Liao et al. used electronic traps connected through ZigBee in a Wireless Sensor Network (WSN), transmitting data via Global System for Mobile Communications (GSM) [17]. Similarly, Philimis et al. developed e-FlyWatch, using traps with image acquisition and GSM/ General Packet Radio Service (GPRS) to send data to a server accessible via a web interface [15]. However, several limitations remain in existing systems. ZigBee-based solutions provide limited communication range and require dense network deployment, making them unsuitable for geographically dispersed orchards. GSM/GPRS-based approaches rely on cellular infrastructure and incur recurring subscription costs, reducing their long-term economic feasibility. In recent smart agriculture applications, both Narrowband Internet of Things (NB-IoT) and LoRa communication technologies have been widely adopted [18]. NB-IoT offers advantages such as higher data rates, extended transmission distance, and relatively low latency [18,19]. However, it typically consumes more power and depends on licensed cellular networks, resulting in higher operational and maintenance costs. In contrast, LoRa is characterized by ultra-low power consumption, lower hardware cost, flexible private network deployment, and minimal recurring fees [18–20]. Given that fruit fly monitoring systems generally transmit small packets of periodic sensor data and do not require high bandwidth or low-latency communication, LoRa is more aligned with the operational requirements of this study. Its long-range coverage and energy efficiency enable large-scale orchard deployment while maintaining cost-effectiveness and extended device lifetime. With LoRa, a wide-area network of traps can connect to a single gateway, simplifying system architecture and allowing gateway placement near stable power and internet sources without relying on solar-powered cellular modules or costly 3G/4G connectivity.

A key element of IoT-based systems is the remote server, which handles data collection, processing, and storage. Most studies on automated fruit fly monitoring use on-premise physical servers and outdated protocols like GSM/GPRS or 3G to connect local stations [15, 16, 21–23]. These technologies are now obsolete in many regions, including Vietnam. Some other studies use cloud-based servers. For example, Miranda et al. combined physical and cloud servers for synchronization and storage [22], while Tsiligiridis et al.'s FruitFlyNet used a coordinator mini-server to send data to the cloud for storage and visualization [24]. Although hybrid systems offer scalability and internet-based access, intermediate servers add complexity and cost. Cloud servers also bring challenges like ongoing fees and limited data control. Furthermore, data security is often under-addressed despite the inherent risks of internet-based systems. Therefore, a research gap remains in the development of a fully integrated, low-power, secure, and scalable fruit fly monitoring platform that combines long-range LoRa communication, lightweight server architecture, device-level authentication, and real-time multi-station management within a unified framework validated under practical field conditions. To address this gap, this study proposes an IoT-based remote management platform for electronic fruit fly traps featuring a customized off-field server with secure remote access, enabling reliable large-scale deployment and improved operational control.

This study presents an IoT-based monitoring platform using smart electronic traps for fruit fly detection. The traps automatically detect and count captured fruit flies, transmitting data to a central gateway via LoRa, which then sends it to a remote physical server over Wi-Fi. This server handles data storage, processing, and also serves as a web server for user access. To ensure secure external access, Cloudflare is used as a protective layer for Hypertext Transfer Protocol (HTTP) / Hypertext Transfer Protocol Secure (HTTPS) traffic [25]. The system collects data on fly counts, energy use, and battery levels, which are periodically updated. A user-friendly web interface displays the number of fruit flies captured and key trap parameters such as battery level and power consumption, helping users perform timely maintenance. The platform is scalable for monitoring other insect pests and can be integrated with early warning systems and automated spraying. It offers strong potential for smart agriculture by supporting early pest detection, informed decision-making, and sustainable resource management.

2. Materials and Methods

2.1 Design of the Trap Network

The electronic trap was developed based on our previously reported smart trap platform, which supports simultaneous real-time acquisition of infrared interruption signals and acoustic wingbeat data [26]. The original system was designed as a multi-sensor monitoring architecture to enhance species discrimination accuracy through data fusion. The present study focuses on validating the reliability of the infrared-based counting method and the remote monitoring capability under field deployment conditions, while the performance of acoustic classification will be investigated in future work. The electronic trap used in this study was developed based on a

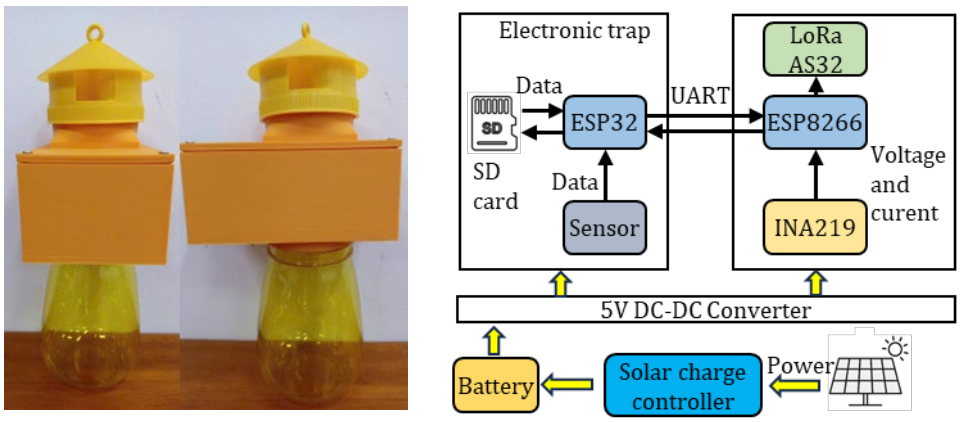
J. of Science and Technology Vol. 18 No. 1 (2026) p. 187-201

commercially available bottle-type traditional trap, as shown in Fig. 1a. The ESP32 microcontroller is used to implement the tracking algorithm and automatically detect fruit flies entering the trap. Specifically, between the lid and the container of the traditional trap is a detection module featuring a centrally located funnel-shaped entry point, which the fruit flies must pass through to enter the container. Along the narrowed section of this funnel-shaped passage, two sets of infrared emitter-receiver LEDs are positioned at the entry and exit points to detect fly movement using a dual-counting interruption mechanism [16, 17].

The double-counting algorithm and its temporal validation strategy were originally developed and experimentally validated in our previous study [27]. In brief, when a fly passes through the funnel, it sequentially interrupts two infrared beams, generating two signal peaks corresponding to the entry and exit sensors. A valid fly entry event is confirmed only when the time interval between these two peaks satisfies the condition $0.08s < t_1 < 100s$. The lower bound (0.08 s) is introduced to eliminate high-frequency fluctuations or noise caused by wing oscillations and transient beam disturbances, while the upper bound (100 s) prevents unrelated or delayed sensor activations from being incorrectly paired as an entry event. Conversely, a fly exit event is confirmed when the time interval between the two peaks satisfies the condition $0.08s < t_2 < 5s$.

Each trap is installed at a station that includes a power supply and a wireless communication module. Fig. 1b illustrates the connection diagram of a station comprising the electronic trap, battery pack, solar panel, and communication circuit between the trap and the gateway. A set of three 2000 mAh 18650 batteries connected in parallel and a 20W solar panel are used to power the entire station. The ESP32 utilizes its dual-core architecture to perform concurrent sensing tasks. Core 0 executes the dual-interruption counting algorithm, continuously monitoring infrared sensor interrupts to detect fly entry events under strict timing constraints. Core 1 is dedicated to high-frequency acoustic data acquisition from the integrated microphone module via the I2S protocol, including buffer management and preliminary signal processing. This configuration is designed to evaluate the ESP32's capability to simultaneously handle real-time event detection and environmental data acquisition from an additional sensor.

Although the ESP32 supports multitasking, the combination of time-critical interrupt handling and continuous audio sampling significantly reduces processing headroom. LoRa communication involves SPI-based packet transmission and handling, introducing non-deterministic latency that may interfere with deterministic sensing performance. In addition, the INA219 current sensor operates via the I2C interface, which competes for GPIO resources already allocated to other peripherals. Therefore, a secondary microcontroller (ESP8266) is employed to manage gateway communication and power monitoring, ensuring stable transmission without compromising real-time detection reliability. Specifically, the ESP8266 collects data from the ESP32, including battery capacity, energy consumption, and the number of fruit flies captured by the trap. It communicates with the ESP32 via UART to retrieve trapping data and manage the operational status of the electronic trap. A LoRa communication module (AS32) is connected to the ESP8266 using the SPI protocol to enable long-range communication with a remote gateway. Power consumption of the station is measured as current using the INA219 current sensor.



(a) **(b)**

Fig. 1 *The electronic trap station: (a) Electronic trap; (b) Block diagram of the electronic trap station*

As previously described, the stations are connected to a gateway responsible for collecting data from each station and transmitting it to the central server. The gateway is designed based on the ESP32 microcontroller and communicates with the stations via the LoRa protocol. It connects to the server through Wi-Fi for data transmission and time synchronization using the Network Time Protocol (NTP) service. Fig. 2 illustrates the

overall architecture of the system. The stations collect data from the traps and transmit it to the gateway via LoRa. The gateway aggregates the data and forwards it to the remote server over a Wi-Fi connection. The data on the server is stored in a database and can be accessed through a web application interface.

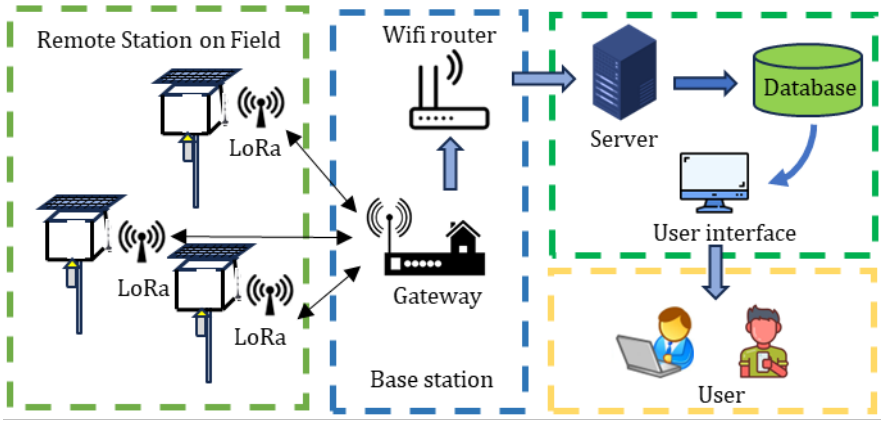


Fig. 2 *Connection diagram of stations, gateway, and server*

2.2 Data Collection and Monitoring Algorithm

During the system's operation, the gateway is responsible for managing the states of all its associated stations, as illustrated in Fig. 3. The gateway periodically checks the real-time clock from the internet and performs scheduled tasks accordingly. First, it wakes up the stations and sends a real-time timestamp including day, month, year, hour and minute to synchronize their clocks. Upon being awakened, the ESP8266 modules at the stations receive the timestamp packets from the gateway. Then, these ESP8266 modules wake up the ESP32 microcontrollers embedded in the electronic traps and forward the synchronized time data. Once the time is synchronized, the ESP32 in each trap creates a new text file to log captured data based on the current timestamp and begins monitoring the number of fruit flies entering the trap.

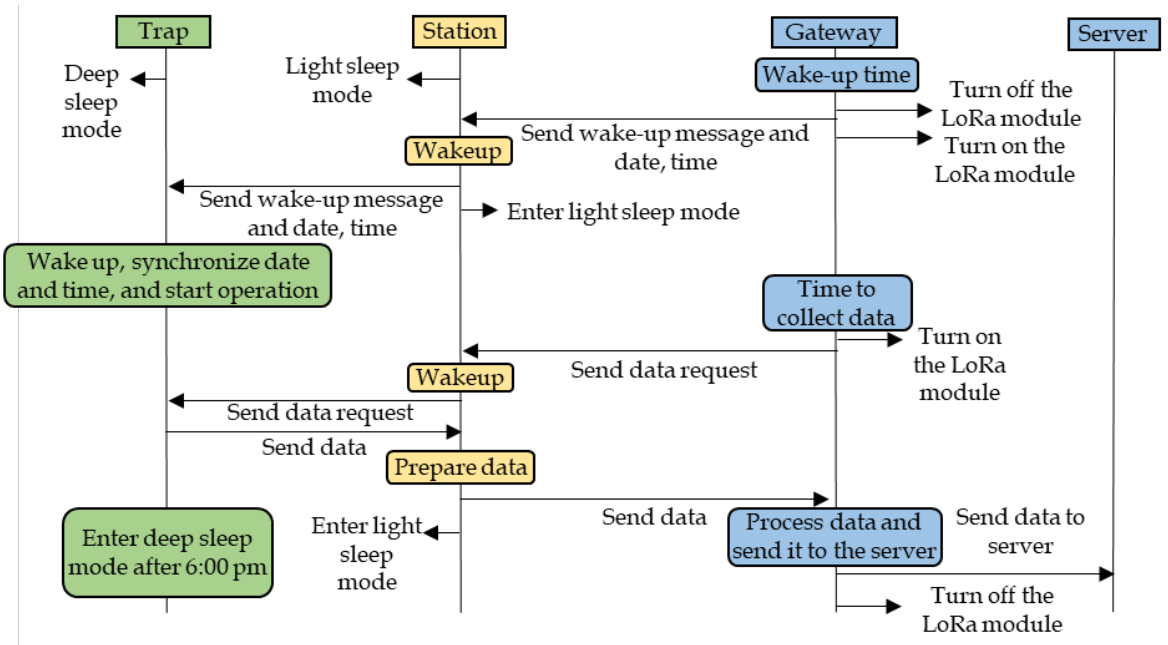


Fig. 3 *System operation diagram*

The second task of the gateway is to periodically request collected data from each station, including battery status, current consumption, and the number of flies captured. Every hour, the gateway sequentially sends a data request to each station. Upon receiving the request, each station responds with its locally stored data. To prevent packet collisions and simultaneous uplink transmissions, the gateway employs a polling-based mechanism in which only one station is addressed at a time. It sends a request and waits for a response within a predefined 20-

J. of Science and Technology Vol. 18 No. 1 (2026) p. 187-201

second timeout window, which is configured to accommodate LoRa transmission latency and packet processing time. The gateway proceeds only after the transaction is completed or the timeout expires. This sequential communication strategy ensures that no two stations transmit data concurrently over the LoRa channel.

If a station fails to respond within the timeout period, the gateway skips it and proceeds to the next one. Once the gateway has completed a full round of requests, it attempts to recontact any stations that previously failed to respond. This retry mechanism is performed up to three times for each unresponsive station, with a one-minute interval between attempts. At 6:00 PM, the gateway issues a final data request, after which the stations automatically enter deep sleep mode based on their internal timers. This cycle repeats daily from 6:00 AM to 6:00 PM. Outside of this time window, only the gateway remains active to periodically update the time.

At the server level, data forwarded by the gateway are processed based on embedded station IDs and timestamps, ensuring correct data association even if multiple gateways transmit to the server simultaneously. Incoming packets are buffered and queued before processing, thereby preventing data overwriting or application-layer conflicts.

Table 1 *Structure of data packets in the system*

Data flow	Structure of packet	Description
Station and trap	hh:mm,dd:mm:yyyy	The string sent from the station to the trap contains information about the date and time
	In:<number>,Out:<number>	The data string transmitted from the trap to the station includes the number of fruit flies captured
Gateway and station	<hh:mm,dd:mm:yyyy,CRC:xx-xx>	The data string transmitted from the gateway includes date and time information
	<ID:xx,Current,Percent,Numflies,CRC:xx-xx>	The data string transmitted from the station to the gateway includes the number of trapped fruit flies and the current energy status of the device
Gateway and server	{"UID": "ESP32-xxx", "Stations": {"Station-<ID>": {"<hh:mm:ss>": {"Current": <mA>, "Percent": <%>, "Number": <num> }}}}	The data string transmitted from the gateway to the server includes information from the stations

The structure of data packets utilized within the system is detailed in Table 1, categorized according to the communication paths: trap and station, station and gateway, and gateway and server. In the first link, the trap and station exchange basic string-based data. The station sends a timestamp to the trap using the format "hh:mm,dd:mm:yyyy". Upon receiving the request, the trap responds with the number of detected events in the format "In:<number>,Out:<number>", where "In" represents the number of insects entering the trap and "Out" indicates the number of insects exiting the trap during the corresponding sampling interval. The direction is determined based on the sequential triggering of the dual infrared sensors. For communication between stations and the gateway, all data strings are encapsulated between the characters "<" and ">", and include a Cyclic Redundancy Check (CRC) code to ensure packet integrity. The gateway sends timestamped requests, while the station responds with its identifier, power consumption (current and battery percentage), and captured fly count. The gateway-to-station packet uses the format <hh:mm,dd:mm:yyyy,CRC:xx-xx>. It is a synchronization request that carries a timestamp generated from the gateway's real-time clock, allowing the station to align its internal clock with a consistent time source. This is essential to ensure uniformity in recorded data across all nodes in the network. The station-to-gateway response packet follows the format <ID:xx,Current,Percent,Numflies,CRC:xx-xx>. It contains the station's unique identifier (ID), the real-time current consumption in milliamperes (Current), the remaining battery capacity as a percentage (Percent), and the number of fruit flies trapped (Numflies). This structured data allows the gateway to monitor not only pest activity but also the energy performance and reliability of each remote station. Finally, the gateway consolidates all collected data and transmits it to the remote server in JSON format, ensuring compatibility and ease of parsing. Each packet includes a unique gateway ID, and nested station data with timestamps and corresponding sensor values.

2.3 Web Server System Design

The server is designed to function as a web server in the fruit fly monitoring system, while also handling data storage and distribution to end-user devices. The system is deployed as a local server on a personal computer running Windows 11 and developed using Node.js (a JavaScript runtime environment) to leverage its asynchronous processing capabilities and scalability. The server operates within a local area network (LAN) and connects to the Internet through Cloudflare Tunnel, which conceals the real IP address, eliminates the need for NAT configuration or a static IP, and still allows access via a fixed domain name. Cloudflare serves as a reverse proxy and dynamic DNS service, enhancing security and simplifying remote access.

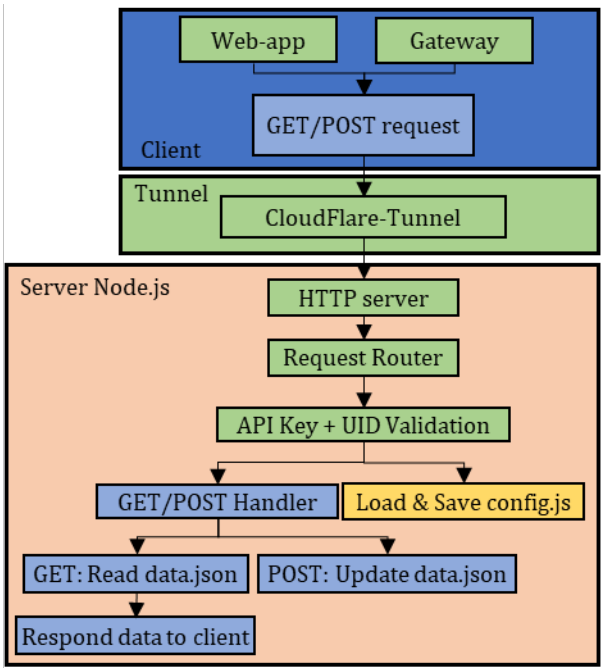


Fig. 4 *HTTP server processing flow*

The server communicates with clients using the HTTP protocol. In the system, clients include both the gateways and the web-based management application. Upon receiving a request from a client, the server performs identity authentication using a custom-built token validation mechanism rather than a standard OAuth framework. Each gateway is assigned a unique identifier (UID) and a corresponding API key during system configuration. These credentials are included in the request header and verified against a registered device list stored on the server. Only requests with matching UID–API key pairs are accepted for further processing; otherwise, they are rejected. After successful authentication, the server processes the request based on the HTTP method (GET or POST). For GET requests, the server queries data from the data.json file and sends it back to the client. For POST requests, the server processes and stores the received data into *data.json*. Additionally, certain configuration-related requests are redirected to a dedicated module for processing and saving into config.js. Fig. 4 illustrates the data flow from the moment a client sends a request to the server's response.

The data collected from the gateways is stored in a tree-structured JSON format, as illustrated in Table 2. In this structure, each gateway represents a top-level element. Within each gateway, information about the gateway itself and the connected stations is included. Each station contains its own metadata as well as operational status details such as energy levels and the number of trapped flies. The data is categorized into two states: real-time information under the "Realtime" key, and daily records under the "dd:mm:yyyy" keys, representing specific dates. For each day, the keys formatted as "hh"mm"ss" hold data for corresponding time points. This design offers several advantages, such as enabling quick data retrieval by gateway, station, or specific timestamp; supporting system scalability by allowing new gateways or stations to be added without altering the existing structure; and optimizing the system for real-time monitoring, data management, and historical analysis.

J. of Science and Technology Vol. 18 No. 1 (2026) p. 187-201

Table 2 *Structure of data storage in the server*

Gateway		Station		Station information	
Key	Value	Key	Value	Key	Value
UID	Unique identifier of the gateway	Station-ID	Unique identifier of each station	Current	Electrical current consumption
Name	Human-readable device name	Name	Station name	Percent	Remaining battery percentage
Type	Device type (e.g., ESP32, Wi-Fi Node...)	Realtime	Key storing the latest data from each station	Number	Number of trapped fruit flies
Position	GPS coordinates of the gateway	Position	GPS coordinates of the stations	"hh:mm:ss"	Timestamp of the collected data (hour, minute, second)
-	-	"dd/mm/yyyy"	Daily historical data (day/month/year format)	-	-

2.4 Study Area and Practical Deployment of the System

The data collection experiment and trap performance evaluation were conducted in a 2,000 m^2 green apple orchard located in Can Tho City, Vietnam (coordinates: 10.116264, 105.659035). During the field test, the electronic traps were automatically activated at 6:00 AM and deactivated at 6:00 PM, as fruit flies are inactive during nighttime. Each evening, the number of trapped flies was also manually counted and recorded. Fig. 5 illustrates the experimental site used for both data collection and field testing.

Fig. 5 *Geographic deployment of stations, gateway, and remote server in the proposed fruit fly monitoring system*

In this study, the system comprised three stations and one gateway. The gateway was placed indoors and connected to Wi-Fi, while two stations were deployed in the orchard, each connected to an electronic trap to collect real-time data, as shown in Fig. 6a and 6b. Fig. 6c presents the fruit flies captured and examined at the end of each day. An additional station was used to simulate sensor data. On the 20th day of testing, a new gateway with three simulated stations was added to the system to evaluate its scalability and ability to integrate new traps. The monitoring application was deployed as a web-based platform at the domain https://flytrap.aiotlab.id.vn, enabling remote access for real-time and historical data monitoring from all gateways and sensor stations. Access is secured through the Cloudflare Tunnel service, which ensures a secure and stable connection without requiring static IP addresses or NAT configuration at the field site.

(a) **(b)** **(c)**

Fig. 6 *A data collection station and an electronic trap: (a) Device arrangement inside a station; (b) Overview of a station with an electronic trap; (c) Fruit flies trapped at the end of the day*

3. Results and Discussion

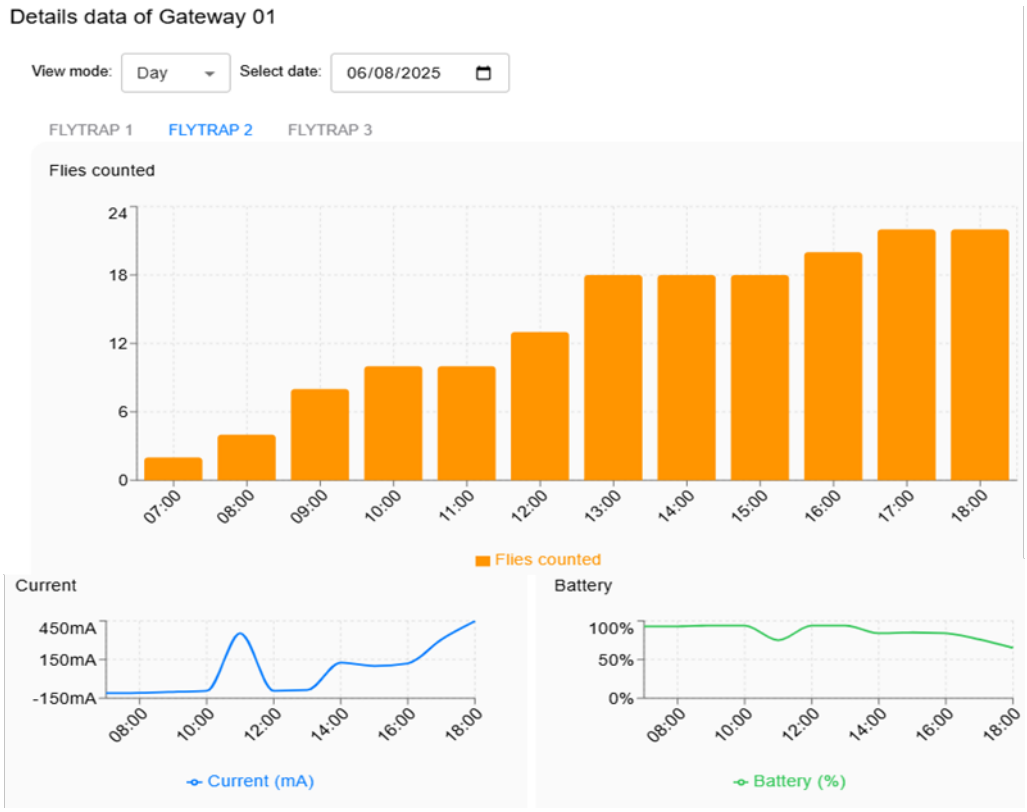
3.1 Evaluation of the Oriental Fruit Fly Monitoring System Design

The system is designed to provide a fully automated solution for monitoring the Oriental fruit fly, suitable for outdoor deployment and scalable for agricultural production. Each monitoring station consists of an insect trap integrated with a pair of infrared sensors to record the number of flies captured. Data is transmitted from the stations to a central gateway, which then relays the information to a remote server via a wireless (Wi-Fi) connection. A key feature of the design is its ability to operate entirely autonomously: each station is powered by a solar-charged battery and optimized for low power consumption, enabling continuous outdoor operation without reliance on the electrical grid. Daily energy level data from the stations indicate that the current configuration ensures uninterrupted operation under sufficient sunlight conditions. However, to maintain functionality during prolonged adverse weather, it is necessary to increase both the battery capacity and the solar panel output.

Fig. 7 illustrates the overview web interface for remotely managing gateways and their connected stations. The interface provides functionalities for monitoring gateway status, displaying recorded fruit fly counts, battery levels, power consumption, and accessing historical data visualizations for each station associated with a given gateway. Users can select and view information for individual gateways by clicking on the corresponding gateway name listed on the left panel. Additionally, clicking on a station icon opens a pop-up window showing detailed station information. The color of each station's circular icon varies according to the number of fruit flies captured, as indicated by the accompanying color bar. A continuous gradient scale is applied, ranging from green (0 flies, low infestation) to yellow/orange (approximately 50 flies, moderate infestation) and red (100 flies or more, high infestation). This visual mapping provides an intuitive alert mechanism when the fly population increases toward critical levels. Furthermore, the "VIEW DETAIL" button allows users to access historical trap data from previous days through interactive charts, as shown in Fig. 8. In this mode, fruit fly data is presented as cumulative bar charts with selectable views by day, month, or year (see Fig. 8a, 8b, and 8c). Additionally, battery levels and current consumption over a single day are visualized using line charts, as shown in Fig. 8a.

J. of Science and Technology Vol. 18 No. 1 (2026) p. 187-201

Fig. 7 *Station management interface*



(a)

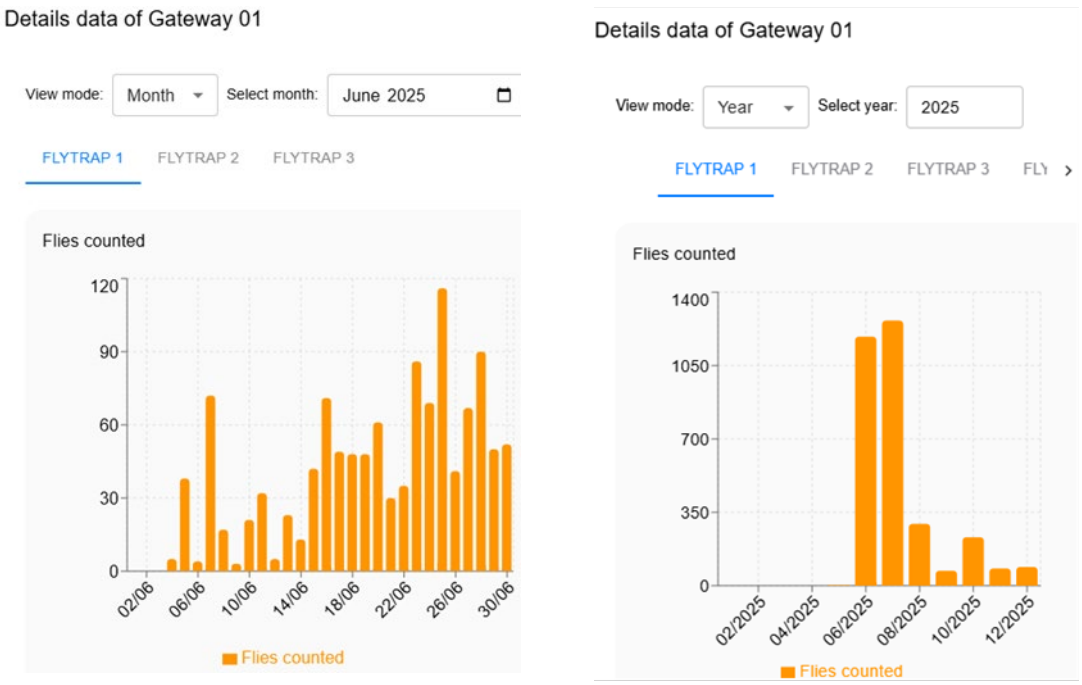


Fig. 8 *Historical data chart: (a) Daily view mode; (b) Monthly view mode; (c) Yearly view*

The entire system is designed in a distributed architecture, with each station functioning as an independent sensing unit, which facilitates future scalability. This design contributes to reduced deployment costs over large areas while ensuring flexibility in maintenance and upgrades. The stable operation of Gateway 2 after being added demonstrates the system's capability to easily expand the number of gateways.

Fig. 9 shows the battery voltage profile over a 48-hour period, starting from 00:00 on the first day to 24:00 on the second day. Two complete charge–discharge cycles were observed. The voltage increased during daytime due to solar charging and gradually decreased at night as a result of system operation. The minimum voltage remained above 3.6 V, which is safely above the Li-ion cutoff threshold (≈3.2 V). No cumulative degradation was observed between the first and second nights, confirming that the 20W solar panel provides sufficient energy to sustain continuous system operation.

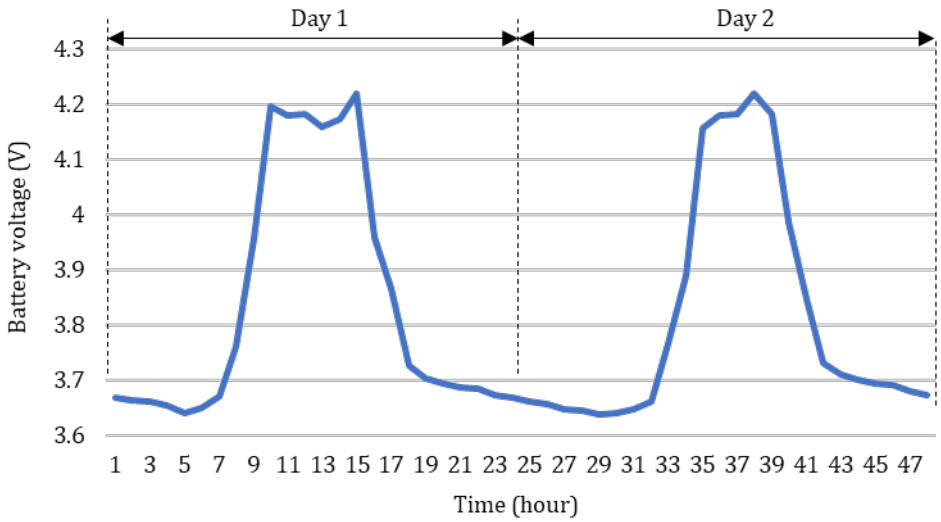


Fig. 9 *Battery charge–discharge cycles over a 48-hour period demonstrating the sufficiency of the 20W solar panel*

J. of Science and Technology Vol. 18 No. 1 (2026) p. 187-201

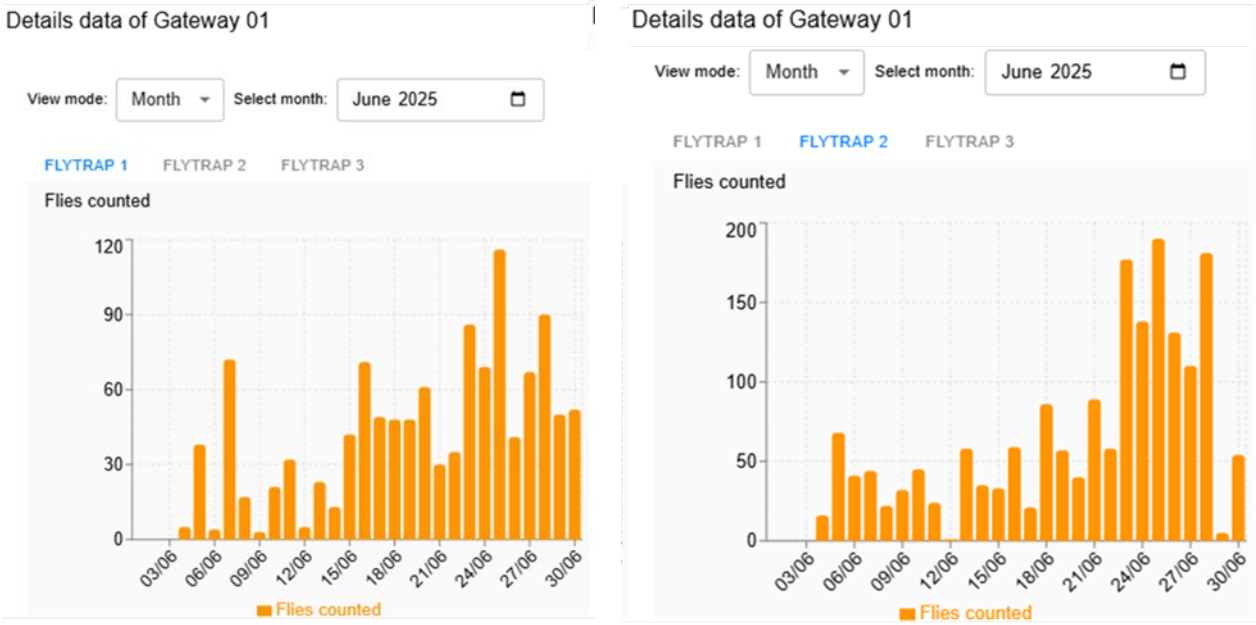
3.2 Data Analysis and Visualization

The system was tested from June 4, 2025, to June 24, 2025. During this period, two monitoring stations installed at the experimental site operated stably and continuously recorded the number of trapped fruit flies over time. The results are illustrated in Fig. 10. Station 1 recorded a total of 773 fruit flies, while Station 2 recorded 1,158 individuals throughout the observation period. The system's accuracy was assessed by comparing sensor data with manual fruit fly counts conducted at the end of each day. The results showed that Station 1 achieved an average accuracy of 86.76%, whereas Station 2 reached 93.50%. To further evaluate the temporal consistency of the system performance, the day-by-day absolute counting error between manual and sensor measurements over the 21-day deployment period is presented in Fig. 11. The results indicate that the error remained relatively stable throughout the experiment, with no observable systematic drift or progressive increase over time. Although occasional peaks were observed on certain days, these fluctuations did not persist and may be attributed to temporary environmental variations or high insect density events. Overall, the system demonstrated consistent counting performance under field conditions.

The difference in accuracy observed between Station 1 and Station 2 may be partially attributed to variations in fruit fly size across locations. Since the infrared sensor detects insects based on beam interruption, smaller individuals may not sufficiently block the beam to be reliably registered, whereas larger insects are more likely to produce clear interruption signals. However, additional environmental and technical factors may also influence detection performance. Variations in ambient light conditions, particularly direct sunlight exposure, may introduce optical interference despite signal modulation. Differences in installation angle, sensor alignment, or minor calibration deviations between stations could further affect sensitivity. Environmental conditions such as temperature and humidity may also impact sensor response characteristics.

In addition, overcrowding may significantly affect counting performance under high-density conditions. When multiple fruit flies simultaneously enter the trap, their bodies may overlap while passing through the sensing region. Because the infrared sensor operates by detecting discrete beam interruption events, such overlap can generate prolonged or merged signals, making it difficult for the algorithm to distinguish individual passages. Consequently, the system may underestimate counts if multiple insects are interpreted as a single event, or occasionally overestimate counts due to signal fluctuations at the beam boundary. Notably, larger deviations were observed on days with elevated insect densities, suggesting that overcrowding contributes to these peak errors.

Future work will involve a more controlled evaluation of environmental variables and sensor calibration consistency, as well as the implementation of adaptive thresholding strategies and the integration of wingbeat sound-based identification using the acoustic sensor to further enhance robustness and counting accuracy under diverse field conditions.



(a) **(b)**

Fig. 10 *Statistics of fruit flies captured during the experimental period: (a) Station 1; (b) Station 2*

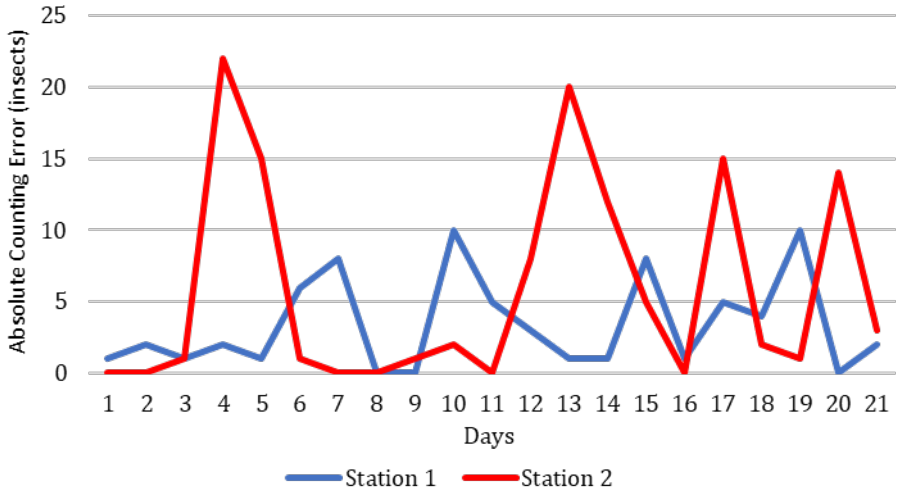


Fig. 11 *Day-by-day absolute counting error between manual and sensor measurements at Station 1 and Station 2 over the 21-day field deployment*

Although some degree of error remains, an accuracy rate exceeding 85% indicates that the system can be effectively applied in practical settings as a replacement for traditional manual counting methods, which are labor-intensive and prone to human error. Moreover, the availability of real-time data and historical access enables farmers and researchers to make timely decisions in pest management.

3.3 Comparison with Related Works

In the field of remote automated fruit fly monitoring systems, previous studies have proposed various approaches to connecting system layers, server types, and database solutions. Table 3 presents a comparison of data transmission methods, server types, as well as storage and security solutions between prior works and the system developed in this study. Earlier research proposed different communication methods among components of remote monitoring systems. For example, Philimis et al. [15] and Jiang et al. [16] employed traditional GSM/GPRS or SMS connections, while Miranda et al. [20] and Bazzi et al. [19] utilized more modern protocols such as ZigBee or 3G/4G in combination with Wi-Fi. These studies demonstrated certain advantages in terms of low power consumption and latency, but faced challenges when scaled to larger deployments or geographically distributed environments. In contrast, the system proposed in this study adopts a hybrid approach, using LoRa technology to transmit data from sensor stations to a gateway, and Wi-Fi/Internet for forwarding the data from the gateway to an off-site server. The use of LoRa enables long-range data transmission with low power consumption, making it well-suited for outdoor operation and long-term deployment without frequent maintenance. Furthermore, relying on LoRa instead of SMS or 3G/4G helps reduce service maintenance costs, improving the system's overall affordability and scalability.

In terms of data storage and processing, Philimis et al. [15] and Miranda et al. [20] employed relational databases (e.g., MySQL, relational geodatabase), while Bazzi et al. [19] adopted PostgreSQL/PostGIS in combination with a web-based interface. However, traditional relational models can sometimes lack the flexibility needed to handle real-time sensor data, which often exhibit heterogeneous or semi-structured formats. In this study, the authors opted to use a NoSQL database, which offers high scalability, fast data retrieval, and native support for semi-structured data, making it well-suited for dynamic environmental monitoring applications.

One key distinction lies in the area of connection security. Most previous studies either did not address or did not clearly implement encryption and security measures. In contrast, this study integrates Cloudflare services to encrypt the entire communication between users and the server via HTTPS/TLS protocols. This ensures data security during transmission over public networks and enhances system availability through protection against DDoS attacks and distributed caching mechanisms.

In summary, the proposed system demonstrates a well-balanced integration of efficiency, scalability, and security, while maintaining low cost and ease of deployment in real-world agricultural environments. However, several limitations of the current system should be addressed in future studies. The system does not yet support automated alert mechanisms, such as notifications sent directly to the manager; the current warning functionality is limited to color indicators displayed on the web app and still requires users to check manually. Additionally, the system lacks advanced data analytics capabilities, such as trend forecasting and density charts. Other features such as user access control and optimization for mobile devices should also be considered to enhance the system's scalability and better meet end-user demands in broader deployment scenarios.

J. of Science and Technology Vol. 18 No. 1 (2026) p. 187-201

Table 3 *Comparison with related studies*

Paper	Connection Method		Server	Database Type	Security Solution
	Station and gateway	Gateway and server			
Philimis et al. [15]	Not mentioned	TCP/IP protocol over GSM/GPRS network	On-premise server	MySQL	Not mentioned
Miranda et al. [20]	ZigBee	Internet	Combination of on-premise server at UIB and cloud server	Relational geodatabase	Not mentioned
Jiang et al. [16]	GSM/SMS	GSM/SMS	On-premise server Host Control Platform	Not mentioned	Not mentioned
Bazzi et al. [19]	Local interface	Wi-Fi / 3G/4G / Zigbee	Off-field server	PosgreSQL, PostGIS + Web	Not mentioned
This study	Lora	Internet	Off-field server	NonSQL database	Cloudflare service used to encrypt communications via HTTPS/TLS protocol

4. Conclusions

This study presented the design and implementation of an automated fruit fly monitoring system that integrates electronic traps with LoRa communication technology and a web-based platform for remote data collection and management. The system utilizes low-power microcontrollers to transmit data to a customized off-field server via Wi-Fi or Internet connection. The management software is deployed as a web interface that supports intuitive data visualization, enabling users to effectively monitor field data. A NoSQL database is employed to provide flexible storage, easy scalability, and optimized performance for non-uniform, time-based sensor data streams. The system also leverages Cloudflare services to enhance the security of data access and system operation. Field experiments conducted over 3 weeks demonstrated relatively high accuracy rates, 86.76% and 93.50% at two testing stations, while also allowing users to conveniently monitor battery levels and current consumption of the devices. The system is low-cost, easy to deploy, and suitable for outdoor use but lacks species classification and has a basic interface. Future improvements include advanced analytics, alert features, and machine learning integration for broader smart agriculture applications. In particular, future work will focus on integrating a fruit fly wingbeat sound recognition algorithm based on frequency-domain features and deep learning models. This enhancement is expected to improve species discrimination and counting accuracy under complex field conditions while maintaining real-time edge deployment capability.

Acknowledgement

The authors acknowledge the financial support from the Ministry of Education and Training of Vietnam under the research project B2024-TCT-21.

The authors used the premium version of QuillBot to assist in grammar checking and language refinement. All content generated was reviewed and verified by the authors, who take full responsibility for the final submission.

Conflict of Interest

The authors declare that there is no conflict of interest regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *Vy-Khang Tran, Van-Khanh Nguyen;* ***data collection:*** *Vy-Khang Tran, Quoc-Duy Tran;* ***analysis and interpretation of results:*** *Vy-Khang Tran, Van-Khanh Nguyen, Chi-Ngon Nguyen;* ***draft manuscript preparation:*** *Vy-Khang Tran, Van-Khanh Nguyen. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] McQuate, G. T., & Liquido, N. J. (2017). Host plants of invasive tephritid fruit fly species of economic importance. *International Journal of Plant Biology Research, 5*, 1072.

[2] Papadopoulos, N. T., De Meyer, M., Terblanche, J. S., & Kriticos, D. J. (2024). Fruit flies: Challenges and opportunities to stem the tide of global invasions. *Annual Review of Entomology, 69*(1), 355–373. https://doi.org/10.1146/annurev-ento-022723-103200

[3] Kibira, M., Affognon, H., Njehia, B., Muriithi, B., Mohamed, S., & Ekesi, S. (2015). Economic evaluation of integrated management of fruit fly in mango production in Embu County, Kenya. *African Journal of Agricultural and Resource Economics, 10*(4), 343–353. https://doi.org/10.22004/ag.econ.211846

[4] Weems, H. V., Heppner, J. B., Nation, J. L., & Fasulo, T. R. (2012). Oriental fruit fly, *Bactrocera dorsalis* (Hendel) (Insecta: Diptera: Tephritidae): EENY-083/IN240. *EDIS, 2012*(3). https://doi.org/10.32473/edis-in240-2012

[5] Zeng, Y., Reddy, G. V., Li, Z., Qin, Y., Wang, Y., Pan, X., Gao, F., & Zhao, Z. H. (2019). Global distribution and invasion pattern of oriental fruit fly, *Bactrocera dorsalis* (Diptera: Tephritidae). *Journal of Applied Entomology, 143*(3), 165–176. https://doi.org/10.1111/jen.12582

[6] Niassy, S., Arimi, J. O., Khamis, F., Etang, D. M., & Ekesi, S. (2022). Insight on fruit fly IPM technology uptake and barriers to scaling in Africa. *Sustainability, 14*(5), 2954. https://doi.org/10.3390/su14052954

[7] Vargas, R. I., Piñero, J. C., & Leblanc, L. (2015). An overview of pest species of *Bactrocera* fruit flies (Diptera: Tephritidae) and the integration of biopesticides with other biological approaches for their management with a focus on the Pacific region. *Insects, 6*(2), 297–318. https://doi.org/10.3390/insects6020297

[8] Hien, N. T., Hendrichs, D., Pereira, R., & Vreysen, M. J. B. (2019). Fruit fly area-wide integrated pest management in dragon fruit in Binh Thuan Province, Viet Nam. In *Area-wide management of fruit fly pests* (pp. 343–347). CRC Press. https://doi.org/10.1201/9780429355738-25

[9] Daniel, C., & Grunder, J. (2012). Integrated management of European cherry fruit fly *Rhagoletis cerasi* (L.): Situation in Switzerland and Europe. *Insects, 3*(4), 956–988. https://doi.org/10.3390/insects3040956

[10] Lello, F., Amiri, M. K. D., Oommen, T. R., Tuma, E., Silva, J. A., & Cruz, T. P. (2023). Fruit fly automatic detection and monitoring techniques: A review. *Smart Agricultural Technology, 1*, 100294

[11] Lima, M. C. F., Leandro, M. E. D. A., Valero, C., Coronel, L. C. P., & Bazzo, C. O. G. (2020). Automatic detection and monitoring of insect pests—A review. *Agriculture, 10*(5), 161. https://doi.org/10.3390/agriculture10050161

[12] Verma, A., & Bodade, R. (2025). Design and implementation of a low-cost IoT-based smart agricultural system using ESP32 and LoRa technology. In *Proceedings of the 2025 International Conference on Computer, Communication and Information Technology (ICCCIT)* (pp. 1–6). IEEE. https://doi.org/10.1109/ICCCIT62592.2025.10928022

[13] Shi, Y., Wang, Z., Wang, X., & Zhang, S. (2015). Internet of things application to monitoring plant disease and insect pests. In *Proceedings of the International Conference on Applied Science, Engineering and Innovation* (pp. 31–34). Atlantis Press. https://doi.org/10.2991/asei-15.2015.7

[14] Molina-Rotger, M., Morán, A., Miranda, M. Á., & Alorda-Ladaria, B. (2023). Remote fruit fly detection using computer vision and machine learning-based electronic trap. *Frontiers in Plant Science, 14*, 1241576. https://doi.org/10.3389/fpls.2023.1241576

[15] Philimis, P., Papadopoulos, R., Tinnirello, I., Vitale, M., & Vreysen, F. R. (2013). A centralised remote data collection system using automated traps for managing and controlling the population of the Mediterranean (*Ceratitis capitata*) and olive (*Dacus oleae*) fruit flies. In *Proceedings of the 1st International Conference on Remote Sensing and Geoinformation of the Environment (RSCy2013)* (pp. 283–290). SPIE. https://doi.org/10.1117/12.2028244

[16] Jiang, J. A., Affognon, H., Muriithi, W., Mohamed, S., & Ekesi, S. (2008). A GSM-based remote wireless automatic monitoring system for field information: A case study for ecological monitoring of the oriental fruit fly, *Bactrocera dorsalis* (Hendel). *Computers and Electronics in Agriculture, 62*(2), 243–259. https://doi.org/10.1016/j.compag.2008.01.005

[17] Liao, M. S., Terblanche, J. S., Papadopoulos, N. T., & Kriticos, D. J. (2012). Development of an autonomous early warning system for *Bactrocera dorsalis* (Hendel) outbreaks in remote fruit orchards. *Computers and Electronics in Agriculture, 88*, 1–12. https://doi.org/10.1016/j.compag.2012.06.008

[18] Sinha, R. S., Wei, Y., & Hwang, S.-H. (2017). A survey on LPWA technology: LoRa and NB-IoT. *ICT Express, 3*(1), 14–21. https://doi.org/10.1016/j.icte.2017.03.004

J. of Science and Technology Vol. 18 No. 1 (2026) p. 187-201

[19] Migabo, E., Djouani, K., Kurien, A., & Olwal, T. (2017). A comparative survey study on LPWA networks: LoRa and NB-IoT. In *Proceedings of the Future Technologies Conference (FTC)* (pp. 29–30). Vancouver, BC, Canada.

[20] Pagano, A., Croce, D., Tinnirello, I., & Vitale, G. (2022). A survey on LoRa for smart agriculture: Current trends and future perspectives. *IEEE Internet of Things Journal, 10*(4), 3664–3679. https://doi.org/10.1109/JIOT.2022.3230505

[21] Bazzi, L., Khan, M. A., Das, D. S., Kader, T. A., & Sharma, V. D. (2022). Fruit fly electronic monitoring system. In *Proceedings of the 15th International Conference on Precision Agriculture*. Minnesota, United States.

[22] Miranda, M. Á., Esposito, G., Silva, R. F., Pereira, L. C., & Santos, J. P. (2019). Developing and implementation of decision support system (DSS) for the control of olive fruit fly, *Bactrocera oleae*, in Mediterranean olive orchards. *Agronomy, 9*(10), 620. https://doi.org/10.3390/agronomy9100620

[23] Jiang, J. A., Njehia, B., Affognon, S., & Kibira, M. (2009). Pest hot spot detection for the oriental fruit fly (*Bactrocera dorsalis* (Hendel)) via wireless sensor networks. In *Proceedings of the Workshop on Wireless Ad Hoc and Sensor Networks* (pp. 1–6).

[24] Tsiligiridis, T., Pontikakos, C., & Perdikis, D. (2014). Architectural issues of a location-aware system applied in fruit fly e-monitoring and spraying control. *AGRIS Online Papers in Economics and Informatics, 6*(4), 195–207.

[25] Cloudflare. (2025). *Connect networks to Cloudflare*. Cloudflare Developers. https://developers.cloudflare.com/cloudflare-one/connections/connect-networks/, Accessed: Mar. 3, 2026.

[26] Nguyen, V. K., Tran, V. K., & Nguyen, C. N. (2025). Towards fruit fly automatic counting: Electronic trap design and long-term feature data acquisition. *International Journal of Applied Science and Engineering, 22*(3), 2025102. https://doi.org/10.6703/IJASE.202509_22(3).004

[27] Nguyen, V. K., Tran, V. K., & Nguyen, C. N. (2026). Cải tiến thuật toán đếm ruồi vàng vào bẫy điện tử dựa trên cơ sở phân tích hành vi. *Tạp chí Khoa học Đại học Cần Thơ, 62*(1), 35–47. https://doi.org/10.22144/ctujos.2026.004