Journal of Applied Informatics and Computing (JAIC)
Vol.x, No.x, Juli 201x, pp. xx~xx
e-ISSN: 2548-6861

Design and Implementation of a Web-Based UAV Telemetry Monitoring System Using Low-Latency WebSocket Data Distribution

Ryan Satria Wijaya [1]*, Jesi Afri Yanti [2]**
* Department of Electrical Engineering Batam State Polytechnic, Batam, Indonesia
ryan@polibatam.ac.id [1], jessyjessy5723@gmail.com [2]

Article Info

Article history:

Received …
Revised …
Accepted …

Keyword:

MAVLink, real-time monitoring, UAV telemetry, web dashboard, WebSocket

ABSTRACT

This research proposes a web-based real-time telemetry monitoring system for unmanned aerial vehicles (UAVs) that employs a WebSocket communication framework to provide concurrent multi-user observation. The MAVLink protocol is employed in the developed system to integrate Pixhawk telemetry collection. The backend architecture utilizing Python and Node.js is employed for telemetry processing, JSON validation, and real-time data transmission. The web-based UAV telemetry monitoring system presents essential flight parameters, including UAV orientation, GPS location, heading, battery status, altitude, speed, and PWM motor signals, via a browser interface that can be accessed simultaneously from many devices. Experimental findings indicate that, during authentic outdoor testing scenarios, the developed platform achieved a WebSocket-based server-to-client telemetry latency of 64.69 ms, calculated from the moment telemetry data was transmitted by the backend processing pipeline to its visualization on the browser dashboard, with latency values fluctuating between approximately 30 ms and 97 ms.

This is an open access article under the CC–BY-SA license.

I. INTRODUCTION

Unmanned aerial vehicles (UAVs) have experienced swift integration into industrial, commercial, and scientific sectors because of their ability to do tasks in perilous or difficult environments that are unsuitable for conventional ground systems. Due to their agility, operational adaptability, and ability to collect real-time information, UAV platforms are gaining significance in fields such as infrastructure assessment, environmental monitoring, aerial cartography, disaster management, and precision farming[1]. Recent advancements in embedded flight controllers, wireless communication, and onboard sensing technologies have markedly improved UAV autonomy and operational reliability[2]. Due to their integration of the effective forward flight attributes of fixed-wing systems with the hovering abilities of rotary-wing aircraft, Vertical Take-Off and Landing (VTOL) platforms have attracted significant research attention across several UAV configurations[3]. VTOL UAVs possess the potential to function in confined spaces without requiring specific runways due to their hybrid flying capabilities[4]. Nonetheless, alternating between forward flight and hovering modes introduces significant control intricacy and requires continuous monitoring of the flight conditions[5].

Telemetry communication is essential in UAV operations since it consistently transmits critical flight information from the onboard flight controller to ground personnel. Telemetry attributes such as attitude data (roll, pitch, and yaw), GPS coordinates, altitude, heading direction, battery status, speed, and propulsion actuator outputs are all essential. These measurements are essential for the safety of aircraft, navigation, and mission oversight as they provide situational awareness[6]. Desktop applications for Ground Control Stations (GCS), such as Mission Planner, server as the cornerstone of contemporary telemetry monitoring systems[7]. Notwithstanding their comprehensive flight management functionalities, these systems exhibit multiple limitations in collaborative operational environments. Initially, it is often essential to install specific software and manually set up the connection port for each client device on conventional GCS platforms. Secondly, concurrent multi-user observation is occasionally constrained by the reality that telemetry access is generally restricted to one monitoring station. These limitations impede adaptability in collaborative

e-ISSN: 2548-6861

scenarios such as multi-operator UAV missions, research exhibitions, and academic flying evaluations[8].

Recent improvements in web technology have enabled the development of browser-based telemetry monitoring solutions that enhance accessibility and eliminate the need for client-side installation[9]. Telemetry information can be promptly exhibited on several devices through standard web browsers due to web-based technology[10]. Nonetheless, several prior systems predominantly focus on basic telemetry visualization and often fail to rigorously evaluate transmission latency, synchronization reliability, and real-time efficacy in practical deployment contexts[11]. Moreover, a significant portion of the existing research has not adequately investigated multi-user synchronization[12].

This research proposes a real-time, web-based UAV telemetry monitoring system utilizing a WebSocket communication framework for low-latency, multi-user telemetry dissemination to address these limitations. The proposed system employs the MAVLink protocol to acquire telemetry information from a Pixhawk flight controller, utilizes a Python backend for processing and validating telemetry packets, and is structured to disseminate synchronized telemetry streams to various connected browser clients through a Node.js WebSocket server. Essential UAV telemetry, encompassing orientation, GPS coordinates, heading, battery status, altitude, velocity, and PWM motor outputs, may be monitored in real-time on the developed dashboard.

II. METHOD

The general methodology utilized to develop and construct the suggested real-time UAV telemetry monitoring system is described in this part. The system architecture that links the UAV platform to browser-based monitoring clients is first described, followed by an explanation of the processing and distribution of telemetry data and a description of the dashboard design and field experiment configuration used to assess system performance.

A. System Architecture

The suggested system structured as a distributed, multi-layer architecture with four primary subsystems as illustrated in Fig 1. The telemetry source, the server system, the link-distribution mechanism, and the client-side dashboard visualization, to achieve low-latency, real-time telemetry delivery from the UAV platform to multiple browser-based monitoring clients[13].

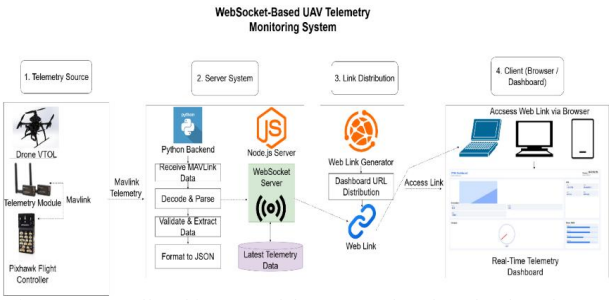

Figure 1. Overall architecture of the proposed WebSocket-based UAV telemetry monitoring system

A VTOL UAV with a Pixhawk flight controller and two wireless telemetry radios serves as the telemetry source. It uses the lightweight MAVLink protocol to transmit flight data, including roll, pitch, yaw, GPS position, altitude, heading, battery condition, airspeed, and motor PWM output, to the ground server[14]. The server system, which consists of a Node.js server and a Python backend, then decodes, validates, and serializes this data into JSON before the Node.js server hosts a WebSocket connection that pushes it to clients instantly. This eliminates the setup overhead of traditional HTTP request–response exchanges and maintains low latency synchronization across multiple users.

Clients can access the monitoring dashboard without installing specific GCS software because link sharing is managed through a shareable dashboard URL created once the WebSocket server is operational. through components like an attitude indicator, a GPS panel, a battery monitor, a speed indication, a compass, and motor PWM bars, several devices' laptops, desktop computers, or smartphones can then open this dashboard simultaneously and get a synchronized data feed. This architecture is lighter, more scalable, and more appropriate for collaborative, multi-user monitoring than desktop-based solutions like Mission Planner since it doesn't require client-side installation.

B. Telemetry Data Processing

The responsibility of transforming unstructured telemetry packets from the UAV into organized data suitable for real-time browser monitoring is assigned to the telemetry data processing phase. This phase encompasses packet reception, packet decoding, telemetry interpretation, data verification, parameter extraction, JSON serialization, and data transmission to the WebSocket server, all executed within the Python backend. Upon receipt of telemetry packets, the wireless telemetry module persistently transmits them to the Python backend[15]. To provide uninterrupted telemetry collection throughout the UAV's flight, the backend maintains a perpetual listening function. The pymavlink library is utilized to interpret the incoming MAVLink packets after their reception. Binary MAVLink packets are interpreted to yield comprehensible telemetry information. The decoded packets are next examined to ascertain the message nature and

extract the telemetry parameters required for real-time observation.

Each decoded packet undergoes verification prior to additional processing to enhance data dependability. To prevent the transmission of incomplete, redundant, or corrupted telemetry data to the monitoring dashboard, the validation process assesses packet integrity and data coherence. When a packet does not pass the validation procedure, it is promptly eliminated. The essential flight parameters, including roll, pitch, yaw, latitude, longitude, altitude, direction, battery condition, flight velocity, and motor PWM outputs, are obtained after successful verification. The extracted telemetry dataset can be expressed as

$$T = \begin{Bmatrix} roll, pitch, yaw, lat, lon, alt, heading, \\ battery, speed, pwm \end{Bmatrix} \quad (1)$$

T represents the aggregation of telemetry metrics transmitted to the surveillance system. Ultimately, to provide seamless interoperability between the Python backend and the Node.js server, the collected telemetry data is encoded in JavaScript Object Notation (JSON) format[16]. Subsequently, the WebSocket server acquires the JSON payload and disseminates it to connected browser clients instantaneously. The processing pipeline improves data integrity and enables low-latency viewing during UAV operations by providing just verified and structured telemetry data.

C. *WebSocket-Based Telemetry Distribution*

The WebSocket protocol facilitates the transmission of telemetry data to the Node.js server for immediate dissemination following its analysis and serialization into JSON format. WebSocket establishes a continuous bidirectional link that remains operational throughout the monitoring session, unlike conventional HTTP communication, which requires repetitive request-response exchanges between the client and server. The server can promptly transmit newly acquired telemetry data to all connected clients without awaiting additional requests due to this communication method[17]. The Node.js server oversees all active WebSocket connections, functioning as the primary hub for communication. Each connected browser client concurrently obtains synchronized flight information due to the telemetry data being transmitted from a singular server instance. When multiple users observe the same UAV operation, this transmission method minimizes communication burden and improves uniformity. The telemetry payload is transmitted in JavaScript Object Notation (JSON) format to enable efficient data interchange. JSON offers a streamlined and platform-agnostic data format that contemporary web browsers can interpret directly without additional transformation. Consequently, when new data is accessible, telemetry visualization can be promptly refreshed, improving system agility during UAV operations. The latency of server-client communication is evaluated utilizing

$$L = T\_receive - T\_send \quad (2)$$

L denotes the total telemetry latency, T_send signifies the moment the server transmits the telemetry packet, and $T\_receive$ indicates the moment the browser client acquires the relevant data.

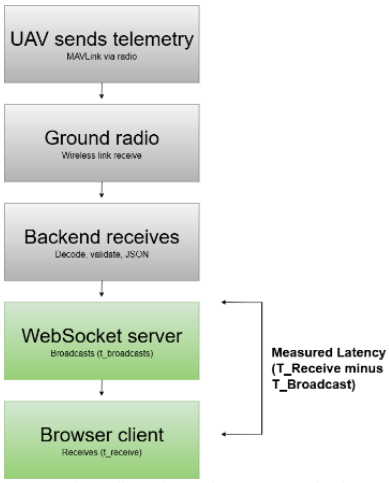

Fig 2. Telemetry receives flow from the UAV to the browser dashboard, showing the broadcast and receiving timestamps used to compute the WebSocket-based distribution latency

Continuous WebSocket connections facilitate low-latency telemetry dissemination to multiple browser clients simultaneously and reduce transmission overhead relative to frequent HTTP polling. During the flight operation, the suggested communication framework maintains data synchronization and visual consistency while facilitating real-time collaborative oversight.

D. *Web-Based Dashboard Design*

The primary platform for observing UAV telemetry in real-time is the web-based dashboard. Users can observe the UAV's status through a browser-based dashboard platform that provides extensive flight data, without the need for specialized Ground Control Station (GCS) software. The interface simultaneously presents many telemetry parameters, as illustrated in Fig 3, allowing operators to observe the UAVs flying status from various perspectives.

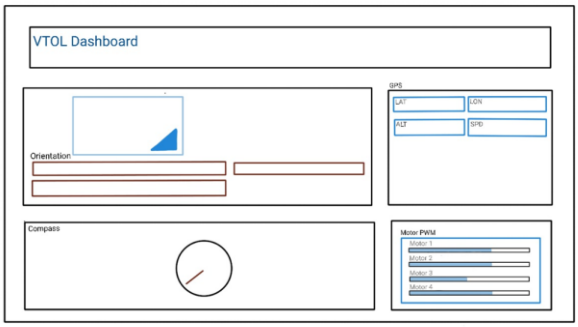

Fig 3. Web-based UAV telemetry monitoring dashboard

e-ISSN: 2548-6861

Upon receipt of each new telemetry packet, the dashboard promptly refreshes all monitoring elements according to the telemetry information transmitted via the WebSocket server. This event driven methodology enhances responsiveness in flight operations by facilitating uninterrupted reading without necessitating frequent client requests or manual page reloads. The user interface has multiple integrated monitoring elements. The artificial horizon displays the roll and pitch angles to indicate the aircraft's orientation. The GPS display presents the latitude, longitude, and altitude information obtained from the flight controller, while a digital compass indicates the present heading. To provide immediate operational insight, designated information displays are utilized to show airspeed, battery status, and flight mode. The dashboard presents motor Pulse Width Modulation (PWM) outputs alongside other telemetry metrics in real-time, in addition to flight status information. These metrics enable operators to identify atypical operational conditions and monitor actuator reactions during flight. The amalgamation of visual instruments and numerical data facilitates quicker comprehension and reduces the time required to assess flying information. The dashboard is reachable from several devices, such as desktop computers, laptops, tablets, and smartphones, provided the device is linked to the same network or has the URL of the generated dashboard, as it functions as a browser-based application. This platform-agnostic method enhances system accessibility and facilitates collaborative multi-user oversight without necessitating additional software installation.

E. Experimental Setup and Performance Evaluation

The proposed UAV telemetry monitoring system underwent empirical evaluation through a field trial to confirm its effectiveness in real-time data acquisition, processing, and visualization. The study aimed to evaluate the complete telemetry communication process, starting with data acquisition at the UAV platform and concluding with real-time display on a web-based dashboard. Figure 4 depicts the experimental setup utilized in the field trials.

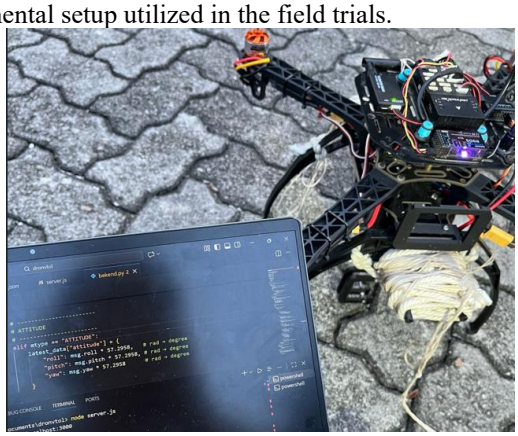

Fig 4. Outdoor experimental setup of the proposed real-time UAV telemetry monitoring system

The experimental setup included a VTOL UAV equipped with a Pixhawk flight controller and a wireless telemetry system. During the flight evaluation, the telemetry unit continuously transmitted flight information to a laptop functioning as the ground station. The laptop simultaneously functioned with the Python backend for MAVLink data processing and the Node.js server for WebSocket telemetry transmission. The telemetry metrics evaluated during the experiment included roll, pitch, yaw, GPS coordinates, altitude, heading, battery status, flight velocity, and motor Pulse Width Modulation (PWM) outputs. The parameters were encoded, authenticated, serialized into JSON format, and transmitted to corresponding browser clients for immediate viewing. The efficacy of communication was assessed by observing telemetry transmission latency and dashboard responsiveness throughout the experiment. The evaluation verified that telemetry updates were reliably aligned across browser clients, maintaining a continuous link between the UAV platform and the monitoring dashboard. The following section showcases experimental results employed to assess the effectiveness of the proposed WebSocket-based telemetry monitoring system.

III. RESULT AND DISCUSSION

Field testing was carried out to validate the operational efficacy, communication performance, and browser-based telemetry viewing features of the suggested UAV telemetry monitoring system. The primary objective of the assessment was to validate the successful amalgamation of the Pixhawk flight controller, MAVLink telemetry connection, Python backend processing, Node.js WebSocket server, and a web-based monitoring dashboard. The various flying parameters, including roll, pitch, yaw, GPS coordinates, altitude, heading, battery status, flight speed, and motor Pulse Width Modulation (PWM) outputs, were consistently documented and shown in real-time during the trials. Additionally, the proposed system's communication efficacy was assessed to determine its ability to provide synchronized oversight among several browser clients and facilitate low-latency telemetry transmission. The subsequent section delineates the experimental results and their corresponding analysis.

A. System Functionality Verification

The proposed UAV telemetry monitoring platform was successfully developed by integrating the Pixhawk flight controller, MAVLink telemetry communication, Python backend processing, Node.js WebSocket server, and a browser-based monitoring dashboard. The developed system operated as a cohesive real-time monitoring platform, enabling telemetry gathering, processing, transmission, and display without requiring specialist Ground Control Station software on each monitoring device.

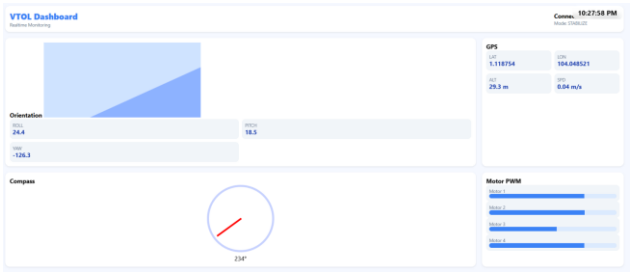

Fig 5. Integrated VTOL monitoring dashboard displaying all telemetry parameters simultaneously

The entire telemetry monitoring dashboard accessible via a browser throughout the field experiment is illustrated in Figure 5. Telemetry data transmitted in real-time from the Pixhawk flight controller over the MAVLink communication protocol was effectively shown on the dashboard. The Python backend interpreted and authenticated each telemetry packet upon its reception by the wireless telemetry module, subsequently transforming the extracted parameters into JSON format. The Node.js server obtained the processed telemetry and employed enduring WebSocket connections to disseminate the data concurrently to all linked browser clients. The successful functioning of this communication chain validates the efficient amalgamation of telemetry acquisition, backend processing, communication, and display into a unified monitoring platform. The developed dashboard provides all necessary flight information for UAV monitoring. The compass display consistently reveals the aircraft's direction, whilst the artificial horizon illustrates the flight's orientation. In addition to battery status, flight velocity, flight mode, and motor Pulse Width Modulation (PWM) outputs, navigational information including latitude, longitude, and altitude is refreshed. As fresh MAVLink packets arrived at the server throughout the aerial experiment, these telemetry parameters were perpetually refreshed, allowing operators to monitor alterations in UAV status with little latency. The proposed WebSocket communication protocol's capability to facilitate uninterrupted real-time data transmission is evidenced by the alignment of telemetry collection and dashboard representation. The browser-centric accessibility of the proposed method is a significant attribute. The established platform enables users to retrieve telemetry data directly via a web browser using the provided dashboard URL, unlike conventional Ground Control Station software that requires specific application installation and communication port setup on every monitoring device. This approach significantly simplifies system implementation in field operations, as multiple operators can simultaneously access the same telemetry data from different devices without the need for extra software installation. Consequently, the proposed architecture maintains synchronized telemetry display across connected clients while enhancing operational adaptability. Upon thorough evaluation, the outcomes of the implementation validate that the proposed UAV telemetry monitoring system successfully fulfills the primary objective of this research, which is to provide a cohesive web-based platform for real-time telemetry oversight. The successful execution establishes a foundation for further evaluations of communication efficacy, system reliability, and telemetry representation quality, all of which are addressed in the subsequent sections.

B. Real-Time Telemetry Visualization Performance

To verify that the proposed monitoring system could reliably display essential UAV flight metrics during the field trial, the efficacy of real-time telemetry visualization was evaluated. The primary focus of the evaluation was the coordination of telemetry gathering, backend processing, WebSocket interaction, and browser visualization. During the testing, the Python backend effectively received telemetry packets dispatched from the Pixhawk flight controller, transformed them into JSON format, and subsequently transmitted them through the Node.js WebSocket server to connect monitoring clients. The flight data on the internet dashboard could be perpetually refreshed with minimal discernible delay due to this communication method.

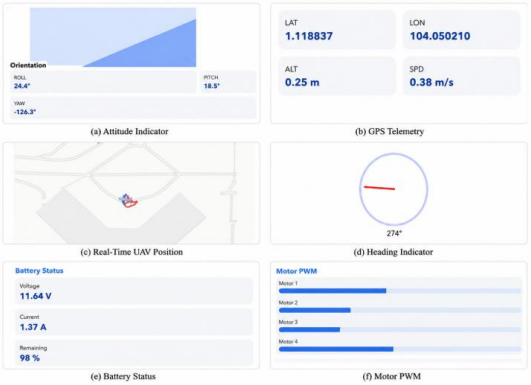

Fig 6. Real-time telemetry visualization during the field experiment: (a) attitude indicator, (b) GPS telemetry, (c) real-time UAV position, (d) heading indicator, (e) battery status, and (f) motor PWM outputs

Figure 6(a) demonstrated that the attitude indicator reliably presents the roll, pitch, and yaw angles of the UAV during its flight. These orientation metrics are essential for evaluating aircraft stability and agility. The continuous updates seen during the trial indicate that the suggested monitoring system proficiently handles attitude data without noticeable disruption, allowing operators to check aircraft alignment instantaneously. The navigational information depicted in Fig. 6(b) and Fig. 6(c) demonstrates the efficient representation of GPS coordinates, elevation, flight velocity, and the present UAV position on the digital map. The correlation between numerical GPS data and the graphical position indicator confirms that navigation telemetry was consistently relayed from the UAV and accurately represented on the browser-based dashboard. This feature provides operators with continuous awareness of the aircraft's position during the flight. The heading data depicted in Fig. 6(d) further substantiates the reliability of telemetry updates. The heading indicator reliably shows the present flight

e-ISSN: 2548-6861

direction based on data supplied from the flight controller. Simultaneously, Fig. 6(e) depicts battery voltage, current usage, and residual battery capacity, enabling operators to monitor the UAV's energy condition throughout the flight. The battery information is essential for preventing unexpected mission interruptions caused by insufficient power. Additionally, Fig. 6(f) illustrates the instantaneous representation of motor Pulse Width Modulation (PWM) outputs. The dashboard continuously updates the PWM values for every motor, allowing operators to observe the control signals generated by the flight controller in different flight scenarios.

The proficient representation of these actuator characteristics confirms that the suggested monitoring system can exhibit both navigation information and fundamental flight control metrics vital for comprehensive UAV oversight. The experimental findings demonstrate that the suggested browser-based monitoring system proficiently visualizes many telemetry parameters simultaneously while maintaining synchronized updates throughout the dashboard elements. The integration of MAVLink connection, Python-based telemetry analysis, Node.js WebSocket delivery, and browser display enables operators to continuously observe UAV status without relying on conventional Ground Control Station software. The findings confirm that the suggested architecture provides an effective resolution for real-time UAV telemetry viewing in collaborative monitoring scenarios.

C. Communication Performance Analysis

A delay evaluated was conducted during the field experiment to examine the communication efficacy of the suggested telemetry distribution framework. In this configuration, the UAV relays flight telemetry via an onboard wireless telemetry radio to a corresponding ground-based telemetry radio linked to the laptop. Upon arrival at the ground station, the unprocessed telemetry stream is transmitted to the Python backend, which interprets, verifies, and converts the data into JSON format prior to relaying it to the Node.js WebSocket server for dissemination to the browser-based dashboard. The latency assessed in this research precisely quantifies the interval from when telemetry data is transmitted by the WebSocket server to when it is received and displayed on the browser client, reflecting the software-related distribution latency of the monitoring pipeline, distinct from the wireless transmission latency between the UAV and the ground station, as previously depicted in Fig 2. This evaluation was performed to determine if the suggested WebSocket-based communication framework can provide sufficiently prompt data dissemination for real-time UAV surveillance applications.

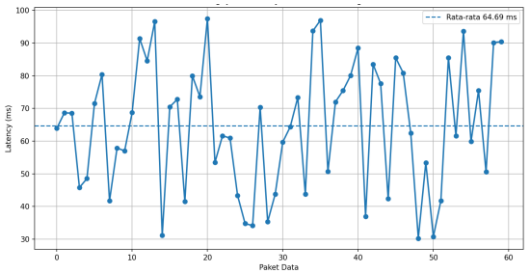

Fig 7. End-to-end telemetry latency of the proposed WebSocket-based monitoring system measured during the field experiment. The dashed line marks the average latency of 64.69 ms

Figure 7 illustrates the latency measured from 60 consecutive telemetry packets throughout the experimental flight test. The latency figures demonstrate variability during the experiment owing to changes in wireless network conditions, packet processing times, and browser rendering performance. Despite these regular fluctuations, telemetry updates continued seamlessly without any interruption in communication. The recorded delay ranged from approximately 30 ms to 97 ms, with an average latency of 64.69 ms, reflecting stable communication efficacy throughout the experiment. The experimental results indicate that the persistent WebSocket connection markedly decreases communication overhead by eliminating the necessity for repeated connection setups typical of conventional HTTP request-response interactions. After interpreting and analysing the telemetry packets with the Python backend, the resulting JSON data was swiftly transmitted via the Node.js WebSocket server to every linked browser client. This communication method enabled coordinated telemetry updates while maintaining minimal delays suitable for real-time UAV oversight.

TABEL I
SUMMARY OF COMMUNICATION PERFORMANCE

Parameter	Value
Communication Protocol	WebSocket
Telemetry Protocol	MAVLink
Data Format	JSON
Number of Telemetry Packets	60
Minimum Latency	30 ms
Maximum Latency	97 ms
Average Latency	64.69 ms

Table 1 summarizes the communication efficacy attained during the field testing. The latency assessments reveal that the suggested monitoring system reliably attained an average response duration of less than 100 ms while handling and transmitting telemetry data to web clients. This communication functionality allows for the ongoing refresh of essential flight metrics, including aircraft orientation, GPS location, heading, battery status, and motor PWM outputs, without noticeable latency from the operator's perspective. The investigation into communication efficacy reveals that

the suggested WebSocket-driven telemetry monitoring framework guaranties dependable and low-latency data transfer for UAV oversight via browsers. The integration of MAVLink telemetry transmission, Python-based telemetry processing, JSON serialization, and WebSocket data distribution enables continuous synchronization between the UAV system and several monitoring clients. The findings demonstrate that the proposed system proficiently enables real-time telemetry oversight and improves accessibility compared to conventional desktop Ground Control Station applications. This evaluation emphasizes the precision and delay of telemetry data provided to a singular monitoring dashboard; testing with multiple browser clients connected concurrently was not performed in this study and is reserved for future research.

IV. CONCLUSION

This document outlines the development and evaluation of a web-oriented UAV telemetry monitoring system that integrates a Pixhawk flight controller, MAVLink communication, Python-driven telemetry processing, a Node.js WebSocket server, and a browser-accessible monitoring dashboard. The suggested framework enables the instantaneous collection, processing, transmission, and display of essential flight information, including aircraft orientation, GPS location, direction, elevation, battery condition, flight velocity, and motor Pulse Width Modulation (PWM) signals. The system utilizes WebSocket connections that inherently provide telemetry to all connected clients, negating the necessity for dedicated Ground Control Station (GCS) software. The developed system effectively displayed telemetry data in real time throughout field trials, as per experimental assessment. The web-based dashboard upheld steady communication efficacy while perpetually exhibiting synchronized flying metrics. The proposed communication framework demonstrates adequate responsiveness for real-time UAV monitoring applications, as the latency assessment indicated an average WebSocket server-to-client distribution latency of 64.69 ms, recorded within the software processing pipeline after telemetry reception at the ground station. These results indicate that the integration of MAVLink, JSON serialization, and WebSocket connectivity enhances monitoring accessible via standard web browsers and guaranties reliable telemetry transmission.

The proposed system offers greater deployment versatility compared to conventional desktop monitoring methods, since it allows monitoring to be conducted effortlessly via a web browser without requiring software installation. Due to this characteristic, the recommended platform is suitable for collaborative oversight scenarios where several operators require access to synchronized telemetry information during UAV operations. Future projects will focus on improving the suggested system by integrating Internet-based telemetry communication, implementing secure user authentication, incorporating telemetry data logging and replay capabilities, and evaluating the system's scalability for multiple UAVs and concurrent monitoring clients. These improvements are expected to increase the utilization of the suggested platform for advanced UAV surveillance and operational oversight.

REFERENCES