International Journal of Marine Engineering Innovation and Research, Vol. 10(4), Dec. 2025. 1124-1132
(pISSN: 2541-5972, eISSN: 2548-1479)

Design and Development of a Command and Control Unit (CCU) on an Unmanned Surface Vehicle (USV) for Water Observation

Setyawan Ajie Sukarno[1*], Hendy Rudiansyah[2], Hasan Tirta Maulana [3], Muhammad Kadir [4], Fajar Setio Adi [5]

(Received: 16 October 2025 / Revised: 16 October 2025 /Accepted: 20 October 2025 / Available Online: 1 December 2025)

***Abstract*— The development of a data acquisition and control system based on modern technologies plays a crucial role in supporting aquatic environmental exploration using Unmanned Surface Vehicles (USVs). This project aims to design and implement a command-and-control unit capable of integrating water survey instruments with microcontroller-based sensors, targeting the real-time display of depth data, GPS location, and graphical visualization on an Smartphone application. The system employs the ESP32 module as the bridge between USV sensors—such as a single-beam echo sounder for depth measurement and a GPS module—and the user application. Wi-Fi connectivity is utilized for fast and efficient data transmission between the USV and the mobile interface. In this project, the Arduino IoT Cloud platform is adopted to simplify the integration of real-time data into the user interface, eliminating the need to develop an application from scratch. The system is designed to be cost-effective, portable, and user-friendly. This project is expected to demonstrate the system's capability to process sensor-acquired data, control the USV's movement direction, and display the results in numerical and graphical formats on the user's Android device in real time.**

Keywords— USV, Data Acquisition, Water Observation, IoT, Command-and-Control Unit

[*]*Corresponding Author: ajie@ae.polman-bandung.ac.id*

I. INTRODUCTION

Earth is a planet where approximately 70% of its surface is covered by water, making water an essential component for sustaining life—not only for humans but also for all living organisms on the planet [1][2]. Major water sources such as rivers, lakes, and oceans provide critical resources for ecosystems. However, not all of these sources maintain water quality suitable for consumption. Various factors, including pollution, climate change, and human activities, have contributed to the degradation of water quality in many regions [3].

Water observation has become a crucial step in assessing water suitability for both human consumption and aquatic ecosystem support. Nevertheless, access to certain water areas remains a significant challenge [4]. Limited geographic information, safety risks, and technological constraints hinder direct observation in remote or hazardous locations [5]. As the importance of preserving aquatic ecosystems grows, so does the need for advanced water monitoring technologies.

Unmanned Surface Vehicles (USVs) offer great potential [6][7] for various applications, including water quality monitoring, bathymetric surveys, military needs, and environmental research [8][9][10][11].

USVs provide an effective solution for data collection in inaccessible or dangerous areas without requiring direct human involvement, thus minimizing risk and enhancing operational efficiency [12][13][14].

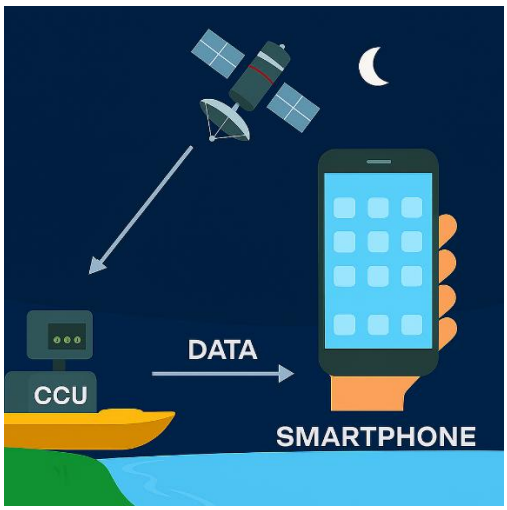

Figure 1. System Overview

In the field of water quality monitoring, traditional approaches such as manual sampling and laboratory testing are time-consuming and do not deliver real-time results. This often leads to delays in detecting critical

Setyawan Ajie Sukarno. Departement of Automation Engineering, Politeknik Manufaktur Bandung, Bandung, 40135, Indonesia. E-mail: ajie@ae.polman-bandung.ac.id
Hendy Rudiansyah. Departement of Automation Engineering, Politeknik Manufaktur Bandung, Bandung, 40135, Indonesia. E-mail: hendy_r@polman-bandung.ac.id
Hasan Tirta Maulana. Departement of Automation Engineering, Politeknik Manufaktur Bandung, Bandung, 40135, Indonesia. E-mail: hasantirta2014@gmail.com
Muhammad Kadir. PT Geotronix Pratama Indonesia, Jakarta, 12440, Indonesia. E-mail: muhammad.kadir@geotronix.co.id
Fajar Setio Adi. PT Geotronix Pratama Indonesia, Jakarta, 12440, Indonesia. E-mail: fajar.adi@geotronix.co.id

changes that may harm the ecosystem [15][16]. With the integration of the Internet of Things (IoT) and modern sensor technologies—such as NodeMCU and various environmental sensors—it is now possible to monitor parameters like pH, temperature, and salinity in real-time via the web or Android devices [17][18].

TABLE 1.
COMPARISON OF TRADITIONAL AND IOT METHOD

Parameter	Traditional Method	IoT-Based Method
Data Collection	Manual sampling	Automated sensors
Real-time Updates	No	Yes
Cost Efficiency	Low	High

Recent studies have demonstrated that combining IoT technologies with USV platforms enables faster and broader data collection, supporting data-driven decision-making for water resource management and environmental disaster mitigation. Moreover, the communication capabilities of USVs via protocols such as Wi-Fi and Bluetooth facilitate efficient data transmission for further analysis [19][20][21]. Prior research has successfully implemented IoT-based USV systems capable of monitoring water quality in real-time by utilizing databases to receive and store sensor data before displaying it on a web interface [22][23].

Generally, conventional survey instruments are not directly compatible with sensors commonly used in microcontroller-based systems due to differences in communication standards. Most survey instruments, whether terrestrial or marine, utilize RS232 for communication and data exchange [24][25]. In contrast, microcontrollers commonly use TTL (Transistor-Transistor Logic), which operates through asynchronous serial communication using UART (Universal Asynchronous Receiver Transmitter) [26] [27][28].

To bridge this gap, the authors developed a Command-and-Control Unit (CCU) for an Unmanned Surface Vehicle that integrates conventional survey instruments with standard microcontroller-compatible sensors. This system aims to enhance the USV's functional capabilities and improve the accuracy of data collection. Furthermore, it is integrated with an Android application for real-time data monitoring, offering users an efficient and accessible solution for water observation tasks [29][30][31].

II. LITERATURE REVIEW

Several previous studies have demonstrated the great potential of utilizing Unmanned Surface Vehicles (USVs) based on the Internet of Things (IoT), integrated with mobile devices, as a more efficient alternative solution. These studies implemented various sensors for water quality, GPS, and Echo Sounders, and employed wireless communication through Wi-Fi, Bluetooth, or the MQTT protocol to transmit real-time data to Android devices. The results indicated that such systems are capable of operating stably and accurately under various environmental conditions, including areas with high pollution levels or adverse weather. This innovation has proven effective for applications such as oil spill detection, water quality monitoring, and bathymetric surveys, while also offering a cost-effective and user-friendly solution.

Based on these findings, this research aims to develop a smartphone-based mobile data acquisition unit as an alternative to conventional survey systems. The designed system is portable and user-friendly, integrating an RS232-based GPS sensor and Echo Sounder with an ESP32 microcontroller, and wirelessly transmitting data via Wi-Fi to a smartphone using the Arduino IoT Cloud platform as the interface. With this approach, the water observation process is expected to be carried out more efficiently, flexibly, and affordably, without compromising the reliability and accuracy of the data obtained.

I. METHOD

This study follows the VDI 2206 design methodology for mechatronic systems, which emphasizes the integration of mechanical, electrical, and software components in a systematic engineering process.

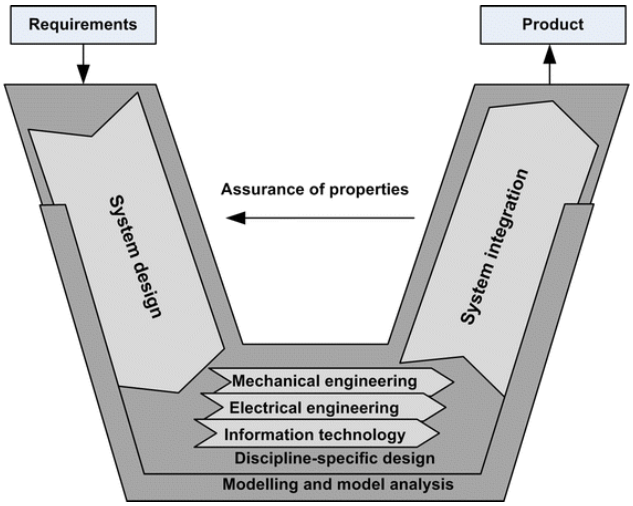

Figure 2. VDI 2206

The development of the Command-and-Control Unit (CCU) for the Unmanned Surface Vehicle (USV) consists of the following phases.

A. Requirement List

The requirements table above is structured to ensure that the system design meets the functional, technical, and environmental needs of the Command and Control Unit (CCU) on the USV. Key requirements such as physical dimensions, RS232 connector compatibility, real-time data acquisition capability, and resistance to vibration and aquatic environments form the foundation of the design process. Additionally, supplementary demands like ease of assembly and system portability are also considered to enable efficient field operations. This list serves as a primary reference for the development of block diagrams and design sketches that follow.

TABLE 2.
REQUIREMENT LIST

Primary Requirements	
Category	Description
Geometry	• Compact, waterproof, and durable • RS232 connector with TTL conversion • RJ45 connector with TTL conversion • Dimensions: 125 mm x 125 mm x 100 mm
Kinematics	• *Real-time* data acquisition • Data wirelessly transmitted to devices
Force	• Resistant to vibrations and shocks • Internal electronics must be protected
Energy	• 5V Power Supply
Signal	• Measurement of depth, temperature, and GPS data • Data stored in database
Material	• Made of corrosion-resistant materials
Additional Requirements	
Assembly	• Lightweight and compact system

B. Block Function Diagram

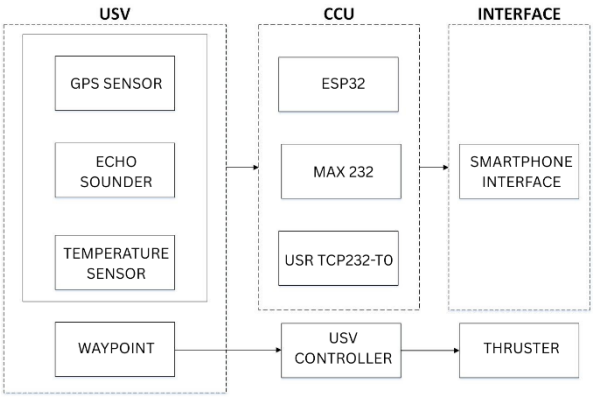

Figure 3. Block Function Diagram

To illustrate the data flow and processing stages more clearly, this figure shows high-level functional diagram of the system based on VDI2206 approach. This diagram highlights how waypoints, sensor data, and user interactions are processed by the microcontroller before being visualized in real time on the smartphone application. The control and processing unit ensures real-time decision making and communication with actuator modules and the user interface.

C. CCU 3D Modelling

The illustration above shows the 3D model of the Command and Control Unit (CCU), designed to visualize the physical layout and internal component placement of the system

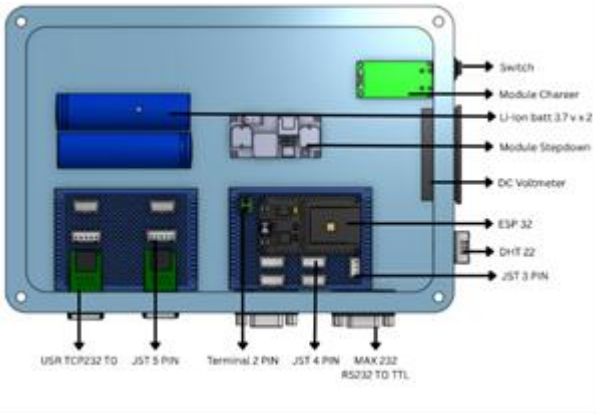

Figure 4. CCU Design

The enclosure is structured to be compact and modular, housing essential components such as:

- Switch: Main power control for the CCU.
- Charging Module: Allows safe recharging of the Li-Ion battery.
- Li-Ion Batteries (2 x 3.7V): Provide power supply for the entire system.
- Step-down Module: Regulates voltage for ESP32 and other modules.
- DC Voltmeter: Monitors system voltage in real time.
- ESP32: Acts as the main microcontroller for processing data and communication.
- DHT22: Sensor for monitoring ambient temperature and humidity.
- JST Connectors (3 to 5 pins): Facilitate modular and secure connections to sensors and modules.
- MAX232 Module: Converts RS232 signals to TTL for microcontroller compatibility.
- USR-TCP232-T0 Module: Converts serial data to TCP/IP for network communication.
- Terminal Blocks: Provide additional wiring flexibility for custom configurations.

This layout ensures organized internal cabling and efficient space utilization, while also protecting electronic components from shock and moisture. The modular connector design (JST) allows for easy integration of external sensors such as GPS and echo sounder modules.

D. Electrical Wiring

The diagram above illustrates the electrical architecture of the Command and Control Unit (CCU), highlighting the interconnection between the main components. At the core of the system is the ESP32 microcontroller, which manages data acquisition, communication, and control. The system is powered by a 7.4V Li-Ion battery, which is charged through a charging module and regulated by a step-down converter to provide stable voltage to the ESP32 and other components. A switch is used to control the power flow to the system.

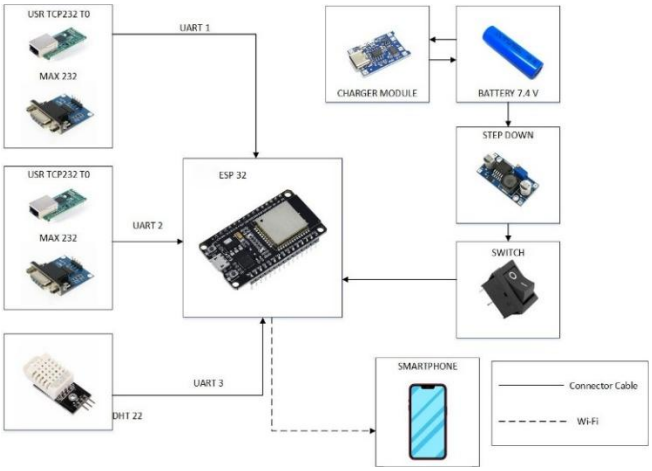

Figure 5. Electrical Wiring

Key sensor and module connections include:

- UART 1: Connects to a USR-TCP232-T0 module and MAX232 level converter for receiving RS232 signals (e.g., from GPS or echo sounder) and converting them to TTL or TCP/IP.
- UART 2: Connects to another MAX232 and USR-TCP232-T0 module for additional serial input (e.g., another survey instrument).
- UART 3: Used for the DHT22 sensor, which measures ambient temperature and humidity.

The ESP32 processes all incoming data and transmits it via Wi-Fi to a smartphone application, where real-time data visualization is displayed. The communication is achieved through the MQTT protocol and integrated with platforms such as Arduino IoT Cloud.

E. System Flowchart

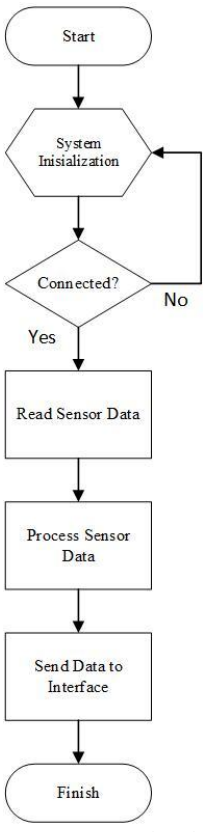

Figure 6. System Flowchart

Figure illustrates the working principle of the system. First, all sensors are initialized so that data can be acquired—such as depth from the Echo Sounder, temperature from the DHT22, and location from the GPS sensor. The data collected from these sensors is processed by the ESP32 by converting the RS232 data format to TTL and formatting it for application use. The processed data is then uploaded to an internal database provided by the Arduino IoT Cloud via Wi-Fi, if an internet connection is available. The stored data is subsequently processed and displayed on a smartphone device. Users can then download the stored data from the database for further analysis and processing.

III. RESULT AND DISCUSSION

A. CCU and Sensor Integration

The integration of the data acquisition unit with the Echo Sounder for obtaining real-time water depth information is shown below.

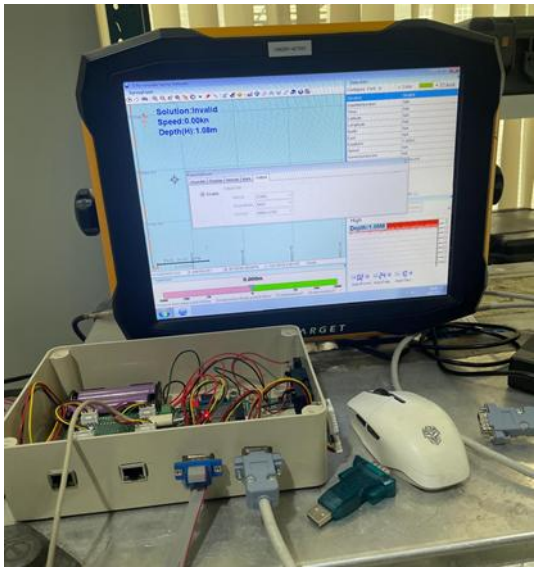

Figure 7. CCU and Echo Sounder Integration

The image above shows the integration between the Mobile Data Acquisition Unit and the Hi-Target HD Lite Single Beam Echo Sounder, which is commonly used for water depth survey measurements and GNSS receiver to get the latitude and longitude from the satellites. The Echo Sounder and GNSS receiver are connected to the unit using a straight RS232 serial cable, with the transducer suspended inside a tank for testing purposes. The SBES output is configured in NMEA 0183 format and transmits the $SDDBT message while $GPGGA message as the output from GNSS receiver. Both of them can be read and processed by the data acquisition unit before being sent to the smartphone interface.

B. Data Accuracy Test

This test was conducted to determine the accuracy of the sensor readings. The test involved comparing the sensor measurements with actual reference data. The sensors are tested by connecting it to the CCU system and Arduino IoT Cloud to collect real-time depth and latitude and longitude data every second. The depth data

compared with the height of the tank while coordinate data from the GNSS receiver compared with coordinate from Google Maps.

TABLE 3.
GNSS RECEIVER TEST

	Sensor Value		Google Maps		
No	Latitude	Longitude	Latitude	Longitude	Error (m)
1	-6,301259	106,781136	-6,301226	106,781056	9,57
2	-6,301263	106,781136	-6,301226	106,781056	9,53
3	-6,301267	106,781136	-6,301226	106,781056	9,49
4	-6,301271	106,781136	-6,301226	106,781056	9,44
5	-6,301276	106,781136	-6,301226	106,781056	9,40
6	-6,30128	106,781136	-6,301226	106,781056	9,36
7	-6,301284	106,781136	-6,301226	106,781056	9,31
8	-6,301289	106,781136	-6,301226	106,781056	9,27
9	-6,301293	106,781136	-6,301226	106,781056	9,23
10	-6,301297	106,781136	-6,301226	106,781056	9,18
11	-6,301301	106,781136	-6,301226	106,781056	9,14
12	-6,301306	106,781136	-6,301226	106,781056	9,10
13	-6,30131	106,781136	-6,301226	106,781056	9,05
14	-6,301314	106,781136	-6,301226	106,781056	9,01
15	-6,301319	106,781136	-6,301226	106,781056	8,97
16	-6,301323	106,781136	-6,301226	106,781056	8,92
17	-6,301327	106,781136	-6,301226	106,781056	8,88
18	-6,301332	106,781136	-6,301226	106,781056	8,84
19	-6,301336	106,781136	-6,301226	106,781056	8,80
20	-6,30134	106,781136	-6,301226	106,781056	8,75
21	-6,301344	106,781136	-6,301226	106,781056	8,71
22	-6,301349	106,781136	-6,301226	106,781056	8,67
23	-6,301353	106,781136	-6,301226	106,781056	8,62
24	-6,301357	106,781136	-6,301226	106,781056	8,58
25	-6,301362	106,781136	-6,301226	106,781056	8,54
26	-6,301366	106,781136	-6,301226	106,781056	8,49
27	-6,30137	106,781136	-6,301226	106,781056	8,45
28	-6,301374	106,781136	-6,301226	106,781056	8,41
29	-6,301379	106,781136	-6,301226	106,781056	8,36
30	-6,301383	106,781136	-6,301226	106,781056	8,32
		Error Average			**8,28**

Error value from the Veripos LD8 GNSS receiver exceeds the tolerance stated in its specification, which is 1.5 meters. This discrepancy may be influenced by the reference data, which was obtained from Google Maps. The accuracy of the comparison would be improved if the sensor data were validated against a predefined benchmark point established specifically for GPS accuracy testing.

The formula used to calculate the error is the Haversine Formula, as follows. To determine the error value from the GPS sensor, the latitude and longitude values from the sensor readings must first be converted into radians:

$$\phi_1 = Sensor\ Latitude \times \frac{\pi}{180}$$

$$\phi_2 = Actual\ Latitude \times \frac{\pi}{180}$$

$$\lambda_1 = Sensor\ Longitude \times \frac{\pi}{180}$$

$$\lambda_1 = Actual\ Longitude \times \frac{\pi}{180}$$

Once the latitude and longitude values are in radians, the distance error can be calculated using the following formula:

$$= sin^2 \frac{(\phi_2 - \phi_1)}{2} + \cos(\phi_1) \times \cos(\phi_2) \times sin^2 \frac{(\lambda_2 - \lambda_1)}{2}$$

$$Error = 2 \times atan2\left(\sqrt{a}, \sqrt{1-a}\right) \times 6{,}371\ x\ 10^6$$

(1)

The coefficient 6.371 × 10^6 represents the Earth's radius in meters.

Using the calculation formula above, the average distance error of the GPS sensor was found to be 8.28 meters.

The water depth test was conducted by installing a Single Beam Echo Sounder (SBES) transducer in a 220-liter test tank with dimensions of 950 mm in height and 590 mm in diameter. The water depth data could be viewed on the dashboard or downloaded as a .csv file from the Arduino IoT Cloud website. The depth data from the SBES was compared to the reference depth, which was calculated by subtracting the draft (the distance from the water surface to the bottom of the transducer) from the total height of the tank. The draft was set at 150 mm, meaning the expected depth measured by the SBES was approximately 800 mm.

TABLE 3.
ECHO SOUNDER TEST

No	Sensor Value	Actual Value	Error (%)
1	4,2	0,8	425,00
2	4,1	0,8	412,50
3	4,1	0,8	412,50
4	4,2	0,8	425,00
5	4,1	0,8	412,50
6	5,6	0,8	600,00
7	5,9	0,8	637,50
8	3,1	0,8	287,50
9	2,4	0,8	200,00
10	3,2	0,8	300,00
11	3,1	0,8	287,50
12	3,4	0,8	325,00
13	2,7	0,8	237,50
14	1,6	0,8	100,00
15	0,0	0,8	100,00
16	1,2	0,8	50,00
17	0,0	0,8	100,00
18	1,9	0,8	137,50
19	1,9	0,8	137,50
20	1,9	0,8	137,50
21	1,9	0,8	137,50
22	1,9	0,8	137,50
23	1,8	0,8	125,00
24	1,8	0,8	125,00
25	1,9	0,8	137,50
26	1,9	0,8	137,50
27	1,9	0,8	137,50
28	1,9	0,8	137,50
29	1,9	0,8	137,50
30	1,9	0,8	137,50
	Error Value		**235,83**

The resulting depth sensor error was calculated to be 235.83%. This indicates that the sensor's accuracy was not within the expected range, as the percentage error was relatively high.

C. Interface Read and Control Test

Sensor readings were tested by sampling 50 data points to verify whether all data were correctly read and stored in the database, as confirmed by their corresponding timestamps.

All data read by the sensors—including the depth sensor, GPS sensor, and temperature sensor—connected to the CCU system via RS232 cables and the TCP/IP protocol, were successfully displayed on the Arduino IoT Cloud interface and properly stored in the database. Therefore, the success rate achieved was 100%.

TABLE 4.
DATA CONTROL TEST

No	Condition	Data	Success Rate
1	ON	Appears	100 %
2	OFF	Stops	100 %
3	ON	Appears	100 %
4	OFF	Stops	100 %
5	ON	Appears	100 %
6	OFF	Stops	100 %
7	ON	Appears	100 %
8	OFF	Stops	100 %
9	ON	Appears	100 %
10	OFF	Stops	100%
Average Success Rate			**100%**

Meanwhile, the data control test evaluated the functionality of the survey switch located on the interface page, which is used to start and stop the display of incoming data on the interface. The test was conducted by toggling the switch on and off five times each via the dashboard.

D. Interface Test

The interface testing was carried out by observing the dashboard on the Arduino IoT Cloud application via a smartphone device. The test was conducted while the smartphone was connected to the internet and the CCU system was connected to the GPS sensor, Echo Sounder, and DHT22. The functionality of the interface could be verified when each section successfully displayed real-time sensor data.

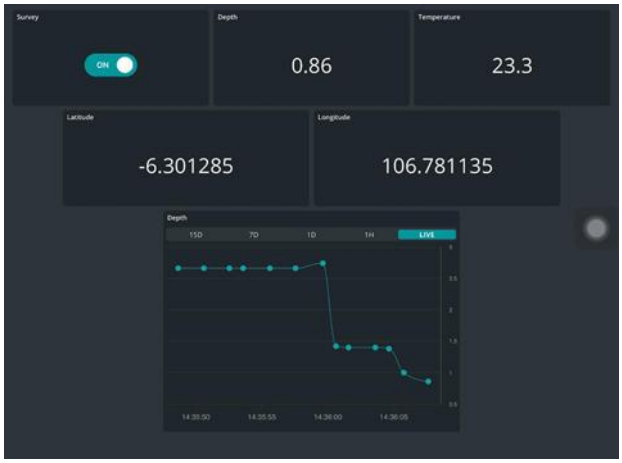

Figure 8. Arduino IoT Cloud Interface

Figure 10 shows the dashboard interface page displaying all data from the sensors connected to the CCU system. When the survey switch was turned on, the interface successfully displayed depth data, air temperature data, and location data represented by latitude and longitude values, formatted according to standard notation. The real-time data graph was also observed to respond dynamically according to the values shown on the depth display. All elements on the interface page functioned properly and were able to display data in real time as expected.

When the hardware is connected to a powered-on CCU system, the ESP32 detects whether there is a Wi-Fi connection matching the programmed credentials. If the correct Wi-Fi is available and connected to the internet, the ESP32 will transmit the sensor data to be displayed on the main page of the IoT Remote application on the smartphone. However, if no Wi-Fi is detected or if Wi-Fi is detected without internet access, the data will not be transmitted to the smartphone and will also not appear on the Arduino IDE serial monitor.

D. Update Rate Test

The Data Update Rate was tested by analyzing sample data downloaded from the Arduino IoT Cloud dashboard. The downloaded .csv file includes timestamps, which can be used to determine the update rate of the CCU system by calculating the average time interval divided by the total number of data points received from various sensors, each with different data transmission frequencies.

TABLE 5.
UPDATE RATE TEST

No	Date	Timestamp	Data
1	03/07/2025	08:23:24.034697228Z	1,8
2	03/07/2025	08:23:25.042938075Z	1,8
3	03/07/2025	08:23:26.073444431Z	1,8
4	03/07/2025	08:23:27.065430103Z	1,8
5	03/07/2025	08:23:28.087454646Z	1,8
6	03/07/2025	08:23:29.170418126Z	1,8
7	03/07/2025	08:23:30.155265694Z	1,8
8	03/07/2025	08:23:31.241230762Z	1,8
9	03/07/2025	08:23:32.233046987Z	1,8
10	03/07/2025	08:23:33.312653597Z	1,8
11	03/07/2025	08:23:34.317995529Z	1,8
12	03/07/2025	08:23:35.323571768Z	1,8
13	03/07/2025	08:23:36.332536208Z	1,8
14	03/07/2025	08:23:37.431449352Z	1,8
15	03/07/2025	08:23:38.412525192Z	1,8
16	03/07/2025	08:23:39.499337826Z	1,8
17	03/07/2025	08:23:40.507633353Z	1,8
18	03/07/2025	08:23:41.512713948Z	1,8
19	03/07/2025	08:23:42.573529681Z	1,8
20	03/07/2025	08:23:43.608579118Z	1,8
21	03/07/2025	08:23:44.678689441Z	1,8
22	03/07/2025	08:23:45.687726232Z	1,8
23	03/07/2025	08:23:46.727385374Z	1,8
24	03/07/2025	08:23:47.72242793Z	1,8
25	03/07/2025	08:23:48.733566007Z	1,8
26	03/07/2025	08:23:49.780630538Z	1,8
27	03/07/2025	08:23:50.762814953Z	1,8
28	03/07/2025	08:23:51.79239974Z	1,8
29	03/07/2025	08:23:52.80384459Z	1,8
30	03/07/2025	08:23:53.810732162Z	1,8
31	03/07/2025	08:23:54.838414485Z	1,8
32	03/07/2025	08:23:55.866244367Z	1,8
33	03/07/2025	08:23:56.881496238Z	1,8
34	03/07/2025	08:23:57.929532596Z	1,8
35	03/07/2025	08:23:58.931036508Z	1,8
36	03/07/2025	08:23:59.963605111Z	1,8
37	03/07/2025	08:24:01.001499284Z	1,8
38	03/07/2025	08:24:02.052584974Z	1,8
39	03/07/2025	08:24:03.04860319Z	1,8
40	03/07/2025	08:24:04.104702136Z	1,8
41	03/07/2025	08:24:05.112637226Z	1,8
42	03/07/2025	08:24:06.131416706Z	1,8
43	03/07/2025	08:24:07.18881802Z	1,8
44	03/07/2025	08:24:08.211150413Z	1,8
45	03/07/2025	08:24:09.284667798Z	1,8
46	03/07/2025	08:24:10.295327663Z	1,8
47	03/07/2025	08:24:11.356835247Z	1,8
48	03/07/2025	08:24:12.366248801Z	1,8
49	03/07/2025	08:24:13.379890613Z	1,8
50	03/07/2025	08:24:14.406520061Z	1,8

Update Rate Formula = Number of Intervals / Total Time (ms)

Start time (row 1) = 08:23:24.034697228

- End time (row 50) = 08:24:14.406520061
- Time difference = 50,372 seconds

Update Rate = 49 / 50,372

= 0,972 Hz

Based on the calculation above, the data update rate of the CCU system is 0.972 Hz, which can be rounded to approximately 1 data point per second. This rate remains consistent regardless of the transmission speed of each sensor.

IV. CONCLUSION

The implementation of a Command and Control Unit for water observation has been successfully realized using a combination of sensors, a NodeMCU ESP32, and supporting electronic modules housed in a single electronic enclosure. The Arduino IoT Cloud platform

was utilized to monitor sensor data and store it in .csv format for further processing. The system demonstrated a relatively high level of accuracy, with the DHT22 temperature sensor achieving 98.65% accuracy. GNSS receiver compared to Google Maps coordinates yielded average positioning errors of 8.28 meters for the Veripos LD8. The Single Beam Echo Sounder tested in a 220-liter tank produced an average error of 235.83%. All sensors showed a 100% success rate in data reading and transmission, and the switch programmed to control data display also operated flawlessly with 100% success. The real-time depth data graph performed reliably in sync with the numeric indicators, without delay or mismatch. The data update rate measured using 50 samples on the Arduino IoT Cloud platform was 0.92 Hz.

ACKNOWLEDGMENTS

The authors would like to express sincere gratitude to PT Geotronix Pratama Indonesia for their valuable support and assistance in the completion of this research.

REFERENCES

[1] J. Agnew, “Borders on the mind: re-framing border thinking,” Ethics Glob. Polit., vol. 1, no. 4, pp. 175–191, 2008, doi: 10.3402/egp.v1i4.1892.

[2] Mangku, D. G. S., Yuliartini, N. P. R., Mercury, S. M., & Darayani, N. M. C. (2022). The Role of the National Agency for Border Management in Maintaining the Territorial Sovereignty in the Land Bord between Indonessia and Timor Leste. Proceeding of 1st Ahmad Dahlan International Conference on Law and Social Justice, 120–131.

[3] S. Ruhana and T. A. Karim, “Indonesia vs. Malaysia: The Battle for Border Territory Resolved”, ILDiSEA, vol. 3, no. 1, pp. 1-32, Jan. 2024.

[4] A. F. Moiz, H. M. Khawaja, R. Sohail, and Z. H. Khan, “QuadSWARM: A real-time autonomous surveillance system using multi-quadcopter UAVs,” in Proc. 2023 Int. Conf. Advanced Innovations in SmartCities (ICAISC-2023). IEEE, 2023.

[5] R. I. H. Abushahma, M. A. M. Ali, N. A. A. Rahman, and O. I. Al- Sanjary, “Comparative Features of Unmanned Aerial Vehicle (UAV) for Border Protection of Libya: A Review,” in 2019 IEEE 15th Int. Colloquium on Signal Processing & Its Applications (CSPA), Penang, Malaysia, Mar. 2019, pp. 114–119. IEEE, doi: 10.1109/CSPA.2019.8695991.

[6] S. A. Sukarno, R. B. Atitallah, and M. Djemai, “Approximation Algorithm for 3-Dimensional Vehicle Routing Problem for Fleet of Multi-Agents,” in 2018 6th International Conference on Control Engineering & Information Technology (CEIT), Istanbul, Turkey: IEEE, Oct. 2018, pp. 1–6. doi: 10.1109/CEIT.2018.8751928.

[7] S. A. Sukarno and Y. Erdani, “Desain Antarmuka Pada Vehicle Routing Problem Untuk Manajemen Armada Multi-Drone,” Jurnal Ilmiah Ilmu Komputer Fakultas Ilmu Komputer Universitas Al Asyariah Mandar, vol. 6, no. 2, pp. 7–14, 2020.

[8] L. Natrayan, S. Kaliappan, and S. Pundir, “Control and monitoring of a quadcopter in border areas using embedded system,” in Proc. Fourth Int. Conf. Smart Electronics and Communication (ICOSEC-2023), 2023, pp. 91–94. IEEE, doi: 10.1109/ICOSEC58147.2023.10276196.

[9] Y. Mekdad, A. Aris, L. Babun, A. El Fergougui, M. Conti, R. Lazzeretti, and A. S. Uluagac, “A survey on security and privacy issues of UAVs,” Computer Networks, vol. 224, 109626, 2023. doi: 10.1016/j.comnet.2023.109626.

[10] S. A. Sukarno, A. Budiyarto, and M. Fadrial, “Pemetaan Citra 3D Untuk Perencanaan Kota Berbasis Fotogrametri Menggunakan Kamera dan Raspberry Pi,” JTT (Jurnal Teknologi Terapan), vol. 10, no. 1, pp. 28–36, 2024.

[11] S. A. Sukarno, H. Rudiansyah, and A. Basyar, “Implementation of Waypoint Navigation and Computer Vision for Monitoring Markers on a Quadcopter Based on ROS (Robot Operating System),” International Journal of Marine Engineering Innovation and Research, vol. 10.

[12] P. Anggraeni, H. Khoirunnisa, M. N. Rizal, and M. F. Alfadhila, “Implementation of WiFi Communication on Multi UAV for Leader-Follower Trajectory based on ROS,” in 2023 Int. Conf. Artificial Intelligence in

[13] Information and Communication (ICAIIC), Bali, Indonesia: IEEE, Feb. 2023, pp. 697–702. doi: 10.1109/ICAIIC57133.2023.10067024.

[14] S. A. Sukarno and R. Ridwan, “Rancang Bangun Datalogger Pada Unmanned Surface Vehicle Untuk Monitoring Posisi Dan Heading Kapal Berbasis Internet of Things,” Jurnal Informatika dan Teknik Elektro Terapan, vol. 12, no. 2, 2024, Accessed: Aug. 21, 2025. [Online]. Available: https://journal.eng.unila.ac.id/index.php/jitet/article/view/4195

[15] H. Khoirunnisa, F. S. Adi, A. Mulyadewi, S. A. Sukarno, Y. Erdani, and P. Anggraeni, “Implementation of IR Lock on Poledrone (Polman Drone Education) for Precision Landing with ROS,” in 2023 IEEE 15th Int. Conf. Computational Intelligence and Communication Networks (CICN), Dec. 2023, pp. 584–590. IEEE.

[16] I. Amiri, A. Shariffuddin, N. Kamel, M. Rahman, M. Bakar, M. B. Mhd Noor, S. Razalli, N. Buniyamin, and M. Khyasudeen, “The development of a GPS-based autonomous quadcopter for precision landing on a moving platform,” Int. J. Vehicle Autonomous Systems, vol. 1, no. 1, pp. 1–18, 2021. doi: 10.1504/IJVAS.2021.10055418.

[17] M. Nugraha, A. Utomo, A. Taufik, R. Tandioga, and R. Syam, “Development of Quadcopter for Tracking Object Using Image Processing,” in IOP Conf. Series: Materials Science and Engineering, vol. 619, pp. 012004, 2019. doi: 10.1088/1757-899X/619/1/012004.

[18] A. Priambodo, F. Arifin, A. Nasuha, M. Muslikhin, and A. Winursito, “A Vision and GPS Based System for Autonomous Precision Vertical Landing of UAV Quadcopter,” J. Phys.: Conf. Ser., vol. 2406, 012004, 2022. doi: 10.1088/1742-6596/2406/1/012004.

[19] L. Tan, X. Lv, X. Lian, and G. Wang, “YOLOv4_Drone: UAV image target detection based on an improved YOLOv4 algorithm,” Computers & Electrical Engineering, vol. 93, 107261, 2021. doi: 10.1016/j.compeleceng.2021.107261.

[20] I. Lebedev, A. Erashov, and A. Shabanova, “Accurate Autonomous UAV Landing Using Vision-Based Detection of ArUco-Marker,” in Interactive Collaborative Robotics. ICR 2020. Lecture Notes in Computer Science, vol. 12336, A. Ronzhin, G. Rigoll, and R. Meshcheryakov, Eds. Cham: Springer, 2020, pp. 279–291. doi: 10.1007/978-3-030-60337-3_18.

[21] M. Aly, “Leveraging Aruco Fiducial Marker System for Bridge Displacement Estimation Using Unmanned Aerial Vehicles,” 2023.

[22] R. Perez-Segui, P. Arias-Perez, J. Melero-Deza, M. Fernández-Cortizas, D. Pérez-Saura, and P. Campoy, “Bridging the Gap between Simulation and Real Autonomous UAV Flights in Industrial Applications,” Aerospace, vol. 10, no. 1, 2023.

[23] H. Qays, B. Jumaa, and A. Salman, “Design and Implementation of Autonomous Quadcopter using SITL Simulator,” Iraqi J. Comput., Commun., Control, and Syst. Eng., vol. 10, 2020. doi: 10.33103/uot.ijccce.20.1.1.

[24] M. H. Li and R. Dayansya, “Trajectory analysis of quadcopter UAV using software in the loop simulation,” in IET Int. Conf. Engineering Technologies and Applications (ICETA 2023), Yunlin, Taiwan, 2023, pp. 200–201. doi: 10.1049/icp.2023.3339.

[25] CMa, Y. Zhou and Z. Li, "A New Simulation Environment Based on Airsim, ROS, and PX4 for Quadcopter Aircrafts," 2020 6th International Conference on Control, Automation and Robotics (ICCAR), Singapore, 2020, pp. 486-490, doi: 10.1109/ICCAR49639.2020.9108103.

[26] Lestari, D., Sujito, Sendari, S., Faiz, M. R., Wang, H. Y., & Prasanta, M. R. (2022). Quadcopter Design with Waypoint Mission Using PID Control System. Proceedings - 11th

Electrical Power, Electronics, Communications, Control, and Informatics Seminar, EECCIS

[27] N. Nair, K. Sareth, R. Bhavani, and A. Mohan, "Simulation and Stabilization of a Custom-Made Quadcopter in Gazebo Using ArduPilot and QGroundControl," in Advances in Robotics and Automation, vol. 1, pp. 205–217, 2022. doi: 10.1007/978-981-19- 0836-1_15.

[28] R. Kumar and S. Jayashankar, "Radar and Camera Sensor Fusion with ROS for Autonomous Driving," 2019 Fifth International Conference on Image Information Processing (ICIIP), Shimla, India, 2019, pp. 568-573, doi: 10.1109/ICIIP47207.2019.8985782.

[29] K. Dang Nguyen and T. -T. Nguyen, "Vision-Based Software-in- the-Loop-Simulation for Unmanned Aerial Vehicles Using Gazebo and PX4 Open Source," 2019 International Conference on System Science and Engineering (ICSSE), Dong Hoi, Vietnam, 2019, pp. 429-432, doi: 10.1109/ICSSE.2019.8823322

[30] S. Gatesichapakorn, J. Takamatsu and M. Ruchanurucks, "ROS based Autonomous Mobile Robot Navigation using 2D LiDAR and RGB-D Camera," 2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP), Bangkok, Thailand, 2019, pp. 151-154, doi: 10.1109/ICA-SYMP.2019.8645984.

[31] A. Mulyanto, R. I. Borman, P. Prasetyawana, and A. Sumarudin, "2d Lidar and Camera Fusion for Object Detection and Object Distance Measurement of ADAS using Robotic Operating System (ROS)," Int. J. Informatics Vis., vol. 4, no. 4, pp. 231–236, 2020, doi: 10.30630/joiv.4.4.466.