Voice-Enabled Emergency Communication for Maritime Workers Using Low-Power LoRaWAN Technology

Zhaniya Zhapparbekova, Simeon Trendov and Dmitry Kachan
Anhalt University of Applied Sciences, Bernburger Str. 57, 06366 Köthen, Germany
zhaniya.zhapparbekova@student.hs-anhalt.de, simeon.trendov@hs-anhalt.de, dmitry.kachan@hs-anhalt.de

Keywords: LoRaWAN, GPS, Raspberry Pi, Arduino, Emergency Communication, Maritime IoT, Low-cost IoT, Search and Rescue (SAR), Telegram Integration.

Abstract: Reliable emergency communication in maritime environments remains challenging due to the limited availability of cellular networks and the high cost of satellite services, particularly for small vessels. To address this gap, a low-cost prototype combining Raspberry Pi 4, LoRa (Long Range) wireless technology, and a GPS (Global Positioning System) module was developed to provide an accessible and energy-efficient emergency alerting solution. The system supports both predefined hardware buttons for specific alerts and a voice-triggered mode in which spoken input is transcribed locally and enriched with real-time geolocation before transmission. Messages are sent via LoRaWAN to an Arduino-based receiver, which forwards them to a Telegram channel for immediate distribution and situational awareness. Laboratory evaluation demonstrated stable operation, acknowledgment-based delivery, and consistent end-to-end transmission delays ranging from 1.3 to 1.8 seconds, confirming the feasibility of integrating multimodal inputs, GPS tagging, and cloud-based messaging within a compact and affordable design. The prototype highlights potential use cases for small fishing vessels, coastal monitoring, inland waterways, and other off-grid safety scenarios in which traditional communication infrastructure is unavailable. Overall, the results illustrate how open-source hardware and lightweight IoT technologies can enable reliable, low-cost emergency communication for maritime workers operating in remote, resource-constrained, or high-risk environments.

1 INTRODUCTION

Reliable communication between vessels and shore-based operators is a critical element of maritime safety. In many nearshore and offshore regions, however, cellular coverage is limited, and satellite services are too costly for small vessels. This gap highlights the need for alternative communication systems that are low-power, affordable, and capable of providing long-range connectivity.

This challenge is particularly evident in developing regions, where local fishermen often travel 10 to 15 km offshore in small boats without access to reliable communication. Lacking affordable satellite phones or subscription services, they are unable to send distress signals during emergencies such as pirate attacks, severe weather, or vessel sinking. These limitations frequently result in preventable accidents and loss of life, underscoring the need for low-cost and robust emergency communication tools.

The LoRaWAN technology provides long-distance wireless communication with minimal energy consumption, making it suitable for such off-grid maritime environments. When combined with GPS, it enables the transmission of geotagged alerts in real time. Through Internet-connected platforms, these alerts can also be forwarded to monitoring centers or rescue teams with minimal delay.

This paper presents the design and implementation of a prototype maritime emergency communication system based on Raspberry Pi 4 and Arduino Mega hardware. The system supports predefined alerts triggered by buttons as well as voice-based input, both automatically enriched with GPS coordinates. Transmitted messages are sent via LoRa to a receiver that relays them to a Telegram channel, ensuring immediate situational awareness.

2 RELATED WORK

Commercial maritime communication systems typically rely on cellular, satellite, or Automatic Identification System (AIS) infrastructures. While cellular links are cost-effective near shore, coverage rapidly degrades offshore; satellite services provide global reach but remain costly for small vessels. AIS supports vessel tracking but does not enable low-latency, user-defined emergency messaging[1].

Recent research has explored low-power wide-area networks (LPWAN) as alternatives for maritime and remote communication. LoRaWAN has been widely evaluated for range, reliability, and limitations across static, dynamic, urban, and free-field scenarios. Trendov et al.[2] demonstrated the development of a LoRaWAN test setup for IoT applications, while subsequent studies assessed its static and dynamic capabilities and confirmed strong long-range performance in free-field environments, though with limitations in dense urban areas[2].

Another important research direction addresses the energy efficiency of LoRaWAN devices. Operational parameters such as spreading factor and bandwidth strongly affect device lifetime[3]. Trendov et al.[4] analyzed the impact of LoRaWAN transceivers on end-device battery lifetime, while complementary work proposed strategies to reduce power consumption under different deployment settings[5]. These insights are crucial for practical, battery-powered emergency systems.

Beyond LoRaWAN benchmarking, several recent studies have focused specifically on maritime contexts[6]. Zhang et al.[7] proposed MCLORA, a maritime ad-hoc communication system based on LoRa, demonstrating feasibility for vessel-to-vessel networking. Ma et al.[8] investigated IoT performance in passenger evacuation scenarios, highlighting the challenges of latency and reliability during emergencies. Liu [9] provided a comprehensive survey on air-to-sea integrated maritime IoT, emphasizing opportunities for combining satellite, aerial, and LoRa-based links. Complementary work by Gong et al. [10] introduced an IoT notification system for marine emergencies that integrates both voice and location-based alerting.

Compared to existing work, the system presented in this paper is distinct in three aspects: (i) it combines multimodal inputs (button- and voice-triggered alerts) on commodity hardware, (ii) integrates GPS to enrich alerts with real-time coordinates, and (iii) forwards them to multiple stakeholders via a widely accessible Telegram channel. This approach addresses both communication range and usability, aiming for a low-cost, replicable solution for small vessels in off-grid environments.

3 HARDWARE SETUP

The prototype was organized into two subsystems: a transmitting unit based on Raspberry Pi 4 and a receiving unit based on Arduino Mega 2560 with a Dragino LoRa/GPS Shield.

The Raspberry Pi 4 was selected for its processing power and native support for Python-based drivers and speech recognition libraries, making it suitable for rapid prototyping in IoT applications [11, 12]. On the receiver side, the Arduino Mega equipped with the Dragino shield acted as a simple LoRa-to-serial forwarder, a configuration commonly used in LPWAN testbeds [2, 13].

This modular separation enabled independent development and debugging of transmission and reception before full system integration.

Table 1: Hardware components of the transceiver unit.

Component	Description
Raspberry Pi 4 Model B	Quad-core ARM Cortex-A72; used as the main controller for LoRa drivers, speech recognition and GPS parsing.
Dragino LoRa/GPS Shield v1.4	LoRa module (SX1276, 868 MHz), connected via SPI and GPIO. GPS functionality was provided instead by an external dedicated module.
Breadboard and jumper wires	Used for prototyping and connecting push buttons with pull-down resistors.
Four push buttons with resistors (10 Ω each)	Configured for predefined alerts (fire, medical, short circuit); one button reserved for voice.
Jabra USB headset	Microphone for voice input; used with the Vosk speech recognition engine.
GPS module (L80-M39, MTK3339 chipset)	Standalone GPS receiver via UART, provides real-time coordinates.

The interconnection between the Raspberry Pi 4, the Dragino LoRa Shield, and the GPS module followed standard Serial Peripheral Interface (SPI) and General-Purpose Input/Output (GPIO) assignments. In particular, MOSI (GPIO10), MISO (GPIO9), SCLK (GPIO11), and CE0 (GPIO8) provided the SPI link, while additional control pins were assigned for `RESET` and `DIO0` interrupts to ensure reliable transceiver initialization and packet handling. The GPS module was connected through the UART interface (TX/RX pins), and push buttons were wired via dedicated GPIO lines with pull-down resistors. A microphone was also attached via USB for speech-based alerts. The wiring and system organization are illustrated in Figures 1–3.

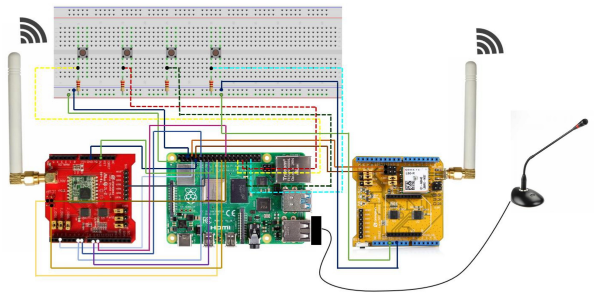

Figure 1: Fritzing wiring diagram of Raspberry Pi 4, LoRa Shield, GPS module, buttons, and microphone.

Raspberry Pi4	LoRa	Breadboard	LoRa/GPS
3.3 V		+	
GND		-	
GPIO 4	D2 (DIO0)		
GPIO 14			RX
GND	GND		
GPIO 15			TX
GPIO 17	D6 (DIO1)		
GPIO 18	D7(DIO2)		
GPIO 27	DIO3		
GPIO 22	D9 (RESET)		
3.3 V	3V3		
GPIO 10 (MOSI)	ICSP 4		
GPIO 9 (MISO)	ICSP 1		
GPIO 11 (SCLK)	ICSP 3		
GPIO 8 (CEO)	D10 (NSS)		
GPIO 5		BUTTON1	
GPIO 6		BUTTON2	
GPIO 13		BUTTON3	
GPIO 19		BUTTON4	

Figure 2: Detailed pin mapping between Raspberry Pi 4 and LoRa/GPS Shield.

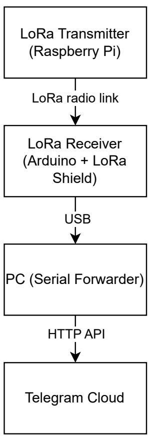

Figure 3: System principle schema. The Raspberry Pi acts as a LoRa transmitter, while the Arduino with LoRa shield functions as the receiver. The Arduino forwards received alerts via USB serial to a PC, which then relays them to the Telegram cloud through the HTTP API.

4 BASELINE PI–LORA–ARDUINO LINK

A baseline communication link was established between the Raspberry Pi transmitter and the Arduino receiver to validate correct wiring and configuration of the SX1276 transceiver. The Raspberry Pi was connected to the Dragino LoRa Shield via the Serial Peripheral Interface (SPI) and selected GPIO lines, following standard SPI conventions (MOSI=GPIO10, MISO=GPIO9, SCLK=GPIO11, CE0=GPIO8). Additional pins, including DIO0, RESET, and NSS, were explicitly assigned for interrupt handling and reliable initialization. On the Arduino side, the LoRa Shield was directly stacked on the Arduino Mega board using its default pin mapping. The overall interconnection is illustrated in Figure 1, which shows the physical wiring.

The software stack was implemented in Python on the Raspberry Pi and in C++ on the Arduino Mega. These lightweight scripts initialized the transceiver, built standardized payloads, and handled acknowledgments. A high-level flowchart (Figure 4) illustrates the end-to-end communication process between the Raspberry Pi transmitter and the Arduino Mega receiver. It highlights how button or voice inputs are converted into LoRa payloads, acknowledged through a stop-and-wait mechanism, and finally relayed to a Telegram channel. This visual representation clarifies the workflow without requiring detailed source code, making the system design easier to interpret.

Finally, a minimal transmitter script was executed to validate the baseline setup. Successful message exchange confirmed interoperability of the two platforms and provided the foundation for subsequent integration of multimodal inputs, GPS enrichment, and Telegram-based cloud forwarding.

The average LoRaWAN packet airtime during prototype testing was approximately 200 ms, and the end-to-end delay from button activation to message appearance in Telegram ranged from 1.3 to 1.8 seconds, depending on acknowledgment retries and radio signal conditions.

5 USER INPUT MODALITIES

Beyond the baseline link, the system was extended with two complementary input modalities. First, three hardware push buttons were connected to the Raspberry Pi to trigger predefined emergency alerts. The first button was fixed as a universal SOS trigger, while the other two were designed as configurable inputs that can be reassigned to different scenarios (e.g., fire, medical emergency, or technical failure). This flexibility allows the system to adapt to different operational contexts without hardware changes, only requiring minor adjustments at the software level.

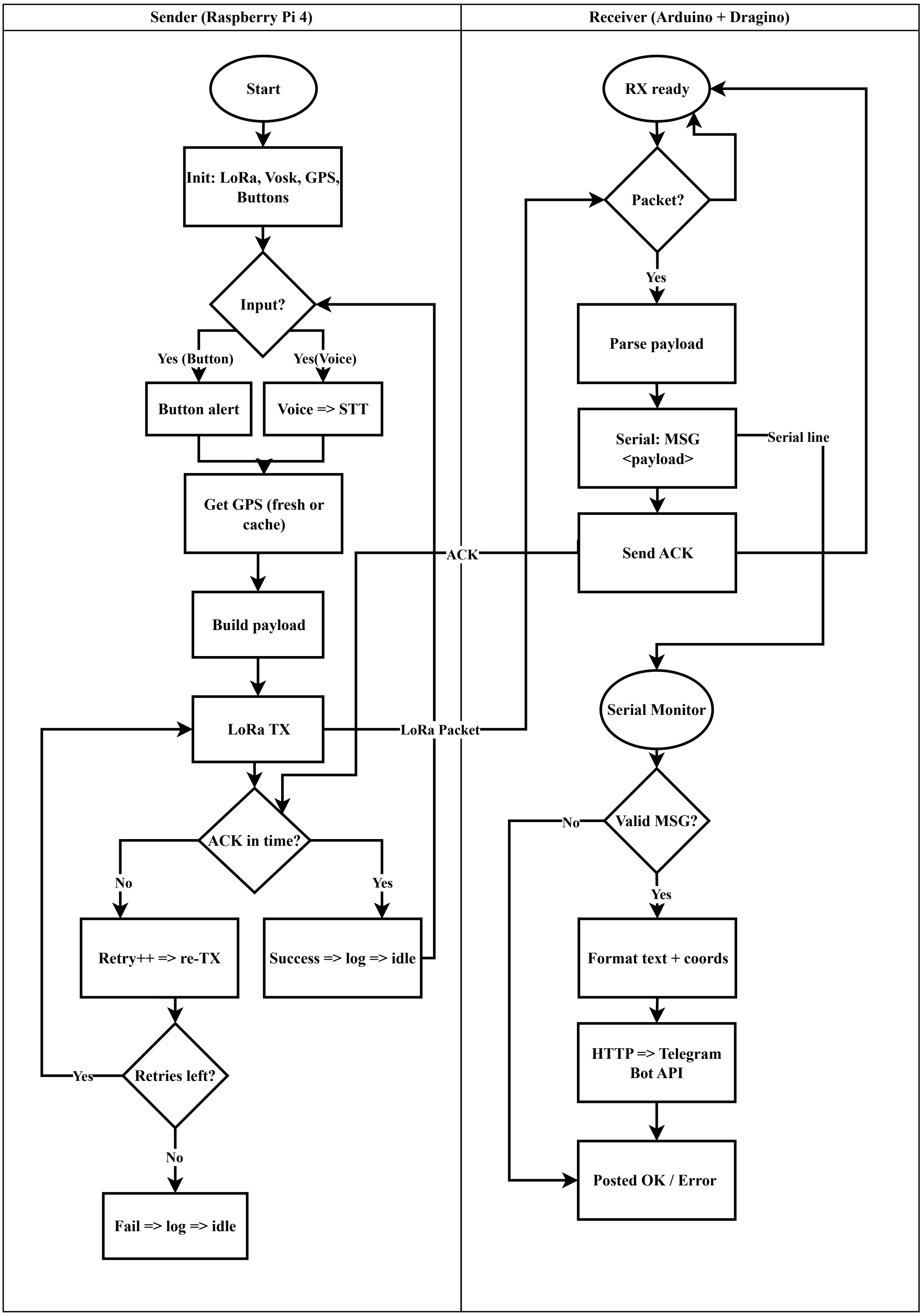

Figure 4: Flowchart of the proposed maritime emergency alert system.

The wiring followed the BCM numbering scheme, as summarized in Table 2. Each button event was handled by a lightweight Python callback, which enqueued the corresponding alert message. Each payload was tagged with a unique vessel ID, allowing receivers and cloud services to identify the origin of the alert. The main loop then built standardized LoRa payloads and transmitted them with acknowledgment (ACK) control. On the receiver side, the Arduino printed each received payload in the format MSG: <VesselID> <payload> and returned a minimal ACK string. This stop-and-wait scheme confirmed message delivery without relying solely on physical-layer CRC, consistent with prior IoT emergency communication designs [14, 15]. Voice-based input was implemented using the Vosk toolkit [16], following similar low-cost speech-enabled IoT prototypes discussed in [14].

Table 2: Push button wiring on Raspberry Pi 4.

Raspberry Pi 4 pin	GPIO	Function
PIN 1 (3v3)	–	Power supply (+rail)
PIN 9 (GND)	–	Ground (–rail)
PIN 29	GPIO 5	Fire alert
PIN 31	GPIO 6	Medical alert
PIN 33	GPIO 13	Short circuit alert

In addition to structured alerts, a voice-based input mode was implemented using a Jabra USB microphone and the Vosk offline speech recognition toolkit [16]. Spoken messages were transcribed locally on the Raspberry Pi and formatted as LoRa payloads in the same way as button-generated alerts, also enriched with the vessel ID for traceability. The workflow can be summarized as: Microphone → Raspberry Pi (audio capture) → Vosk (speech-to-text) → LoRa transmission.

Both button and voice inputs followed the same sender workflow described in Figure 4. This integration significantly enhanced the system's flexibility in real-world emergency contexts while keeping the underlying LoRa driver unchanged.

6 SYSTEM EXTENSIONS

Beyond the baseline link and button/voice input, the system was extended with geolocation and cloud integration to provide richer situational awareness and global reach. These two features—GPS tagging and Telegram-based message relay—transform the prototype from a laboratory setup into a practical maritime emergency tool.

GPS integration. A standalone GPS module (L80-M39, MTK3339 chipset) was connected to the Raspberry Pi 4 via the UART interface. The module's TX pin was wired to the Pi's RX (GPIO15), its RX to the Pi's TX (GPIO14), while power and ground were supplied from the 3.3 V and GND rails. This standard configuration allowed the Pi to continuously receive NMEA sentences directly from the GPS. The LoRa driver was extended with the `pynmea2` library to parse these messages and automatically attach geographic tags to all outgoing alerts. To improve robustness, the most recent valid coordinates were cached and included even if fresh satellite fixes were temporarily unavailable. By embedding real-time geolocation into each alert, the system enabled responders to both classify incidents and immediately locate distressed vessels, a critical improvement for maritime search-and-rescue operations [17].

Telegram integration. While LoRa provides reliable local transmission, its range is inherently limited to tens of kilometers in open areas [18]. To enable global situational awareness, the system was integrated with Telegram Messenger using its public Bot API [19], similar to recent IoT-based emergency alerting frameworks [10, 7].The forwarding chain followed a modular three-step design (Figure 5):

1) LoRa transmission (Raspberry Pi → Arduino): the Raspberry Pi prepared alert payloads <VesselID> <coords> <message>, which were received by the Arduino via a Dragino LoRa Shield.
2) Serial forwarding (Arduino → PC): the Arduino printed received payloads to its serial port in the format MSG: <payload>.
3) Telegram relay (PC → Cloud): a lightweight Python script monitored the serial port and relayed valid messages to a preconfigured Telegram channel via HTTPS requests.

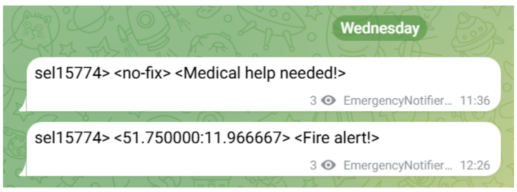

Figure 5: Example of emergency message delivery to a Telegram channel.

Downlink option. Although the present prototype implements only uplink transmission from vessel to shore, an important future extension is the addition of a downlink channel. This would allow authorities or rescue centers to send short control messages back to the vessel. For instance, a Telegram command entered by an operator could be relayed from the cloud to the PC, forwarded via the Arduino through LoRa, and received by the Raspberry Pi onboard. Such a feature would transform the system into a bidirectional communication tool, enabling interactive coordination and enhancing overall maritime safety [20].

This modular separation allowed the Arduino to function as a transparent LoRa-to-Serial bridge, while the host PC provided Internet-based message delivery. By relaying alerts through Telegram, emergency notifications generated onboard could be disseminated in real time to multiple stakeholders, ensuring global accessibility wherever Internet connectivity is available.

7 CONCLUSIONS

This paper presented a modular emergency communication system that integrates LoRaWAN, Raspberry Pi, Arduino Mega, GPS, and Telegram to enable reliable alerts in off-grid maritime environments. The solution directly addresses the communication gap faced by small-scale fishermen in regions such as Trinidad and Tobago, where limited cellular coverage and the high cost of satellite phones leave crews without affordable means of requesting assistance in emergencies. This system can also be scaled and adapted to other maritime regions worldwide.

The prototype was developed incrementally, introducing hardware buttons, voice input, and GPS geotagging before adding a cloud relay via Telegram. Laboratory validation confirmed reliable LoRaWAN transmission with acknowledgments, while multimodal inputs and GPS ensured both flexibility and situational awareness. Similar approaches have been applied in other maritime IoT studies [8, 9], confirming the relevance of combining LPWAN, GPS, and cloud integration for safety-critical contexts.

Beyond small fishing vessels, the system can be adapted to other remote applications such as disaster response, rural monitoring, or inland waterway transport, where low-cost and resilient communication is essential. An important future extension is the implementation of a downlink option, allowing authorities to transmit control or coordination messages back to vessels through Telegram and LoRa.

By combining open-source hardware, lightweight IoT protocols, and cloud services, the results highlight a practical pathway toward enhancing maritime safety in off-grid regions. Future work will focus on extended field trials, optimization of power consumption, and integration of additional security mechanisms.

ACKNOWLEDGEMENT

We acknowledge support by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) and the Open Access Publishing Fund of Anhalt University of Applied Sciences.

REFERENCES

[1] S. Pensieri, F. Viti, G. Moser, S. B. Serpico, L. Maggiolo, M. Pastorino, D. Solarna, A. Cambiaso, C. Carraro, C. Degano, I. Mainenti, S. Seghezza, and R. Bozzano, “Evaluating LoRaWAN Connectivity in a Marine Scenario,” Journal of Marine Science and Engineering, vol. 9, no. 11, p. 1218, 2021. [Online]. Available: https://doi.org/10.3390/jmse9111218

[2] S. Trendov, B. Zadoienko, E. Siemens, D. Kachan, M. Kalendar, M. Gering, and S. Maksymov, “Evaluation of LoRaWAN’s static and dynamic capabilities and its limitations for IoT applications,” Proc. Int. Conf. on Applied Innovation in IT, vol. 11, no. 1, pp. 7–16, 2023, doi: 10.25673/101901.

[3] B. Miles, E.-B. Bourennane, S. Boucherkha, and S. Chikhi, “A study of LoRaWAN protocol performance for IoT applications in smart agriculture,” Computer Communications, vol. 164, pp. 148–157, 2020. [Online]. Available: https://doi.org/10.1016/j.comcom.2020.10.009

[4] S. Trendov, E. Sariiev, K. B. S. Mukhtar, D. Kachan, and E. Siemens, “Lpwan technologies range evaluation: LoRaWAN, NB-IoT, Sigfox, and LTE-M in urban and free-field settings,” in Proc. 7th Global Power, Energy and Communication Conf. (GPECOM), Bochum, Germany, 2025, pp. 1089–1094, doi: 10.1109/GPECOM65896.2025.11061914.

[5] S. Trendov, E. Stoilkovska, and E. Siemens, “Impact of LoRaWAN operational parameters on energy efficiency and ways to improve it,” in Information and Communication Technologies and Sustainable Development (ICT&SD 2022), ser. Lecture Notes in Networks and Systems, S. Dovgyi, O. Trofymchuk, V. Ustimenko, and L. Globa, Eds. Cham: Springer, 2023, vol. 809, doi: 10.1007/978-3-031-46880-3_10.

[6] J. Pinelo, A. D. Rocha, M. Arvana, J. Gonçalves, N. Cota, and P. Silva, “Unveiling LoRa’s Oceanic Reach: Assessing the Coverage of the Azores LoRaWAN Network from an Island,” Sensors, vol. 23, no. 17, p. 7394, 2023. [Online]. Available: https://doi.org/10.3390/s23177394

[7] J. Zhang et al., “Maritime ad-hoc communication system based on lora,” Ocean Engineering.

[8] Y. Ma et al., “Iot performance for maritime passenger evacuation,” Sensors, vol. XX, 2024.

[9] S. Liu, “A survey on air-to-sea integrated maritime internet of things,” Journal of Marine Science and Engineering, vol. 12, no. 1, p. 11, 2023, doi: 10.3390/jmse12010011.

[10] D. H. Gong et al., "Iot notification system for marine emergencies with voice and location alerting," in Proc. Int. Journal of Internet, Broadcasting and Communication, 2022.

[11] S. Monk, Programming the Raspberry Pi: Getting Started with Python. McGraw-Hill, 2019.

[12] M. Barr, Programming Embedded Systems in C and C++. O'Reilly Media, 1999.

[13] S. Trendov, E. Sariiev, K. B. S. Mukhtar, D. Kachan, and E. Siemens, "Comparison of performance and power consumption in sigfox, nb-iot, and lte-m," in Applied Innovations in Information and Communication Technology. ICAIIT 2024, ser. Lecture Notes in Networks and Systems, S. Dovgyi, E. Siemens, L. Globa, O. Kopiika, and O. Stryzhak, Eds. Cham: Springer, 2024, vol. 1338, pp. 127–158, doi: 10.1007/978-3-031-89296-7_8.

[14] C. R. Panigrahi et al., "Low-cost iot-enabled voice communication systems for emergency scenarios," IEEE Access, vol. 9, pp. 12 345–12 356, 2021.

[15] M. Z. A. Bhuiyan et al., "Internet of things (iot) and cloud computing enabled disaster management — a survey," Future Generation Computer Systems, vol. 79, pp. 306–324, 2018.

[16] "Vosk speech recognition toolkit," [Online], Alpha Cephei Inc., available: https://alphacephei.com/vosk.

[17] "Global maritime distress and safety system (gmdss) manual," International Maritime Organization, 2019.

[18] "Lorawan regional parameters eu868," [Online], LoRa Alliance, 2020, available: https://lora-alliance.org.

[19] "Bot api documentation," [Online], Telegram, available: https://core.telegram.org/bots/api.

[20] M. Lyimo, B. Mgawe, J. Leo, M. A. Dida, and K. Michael, "Adaptive Antenna for Maritime LoRaWAN: A Systematic Review on Performance, Energy Efficiency, and Environmental Resilience," Sensors, vol. 25, no. 19, p. 6110, 2025. [Online]. Available: https://doi.org/10.3390/s25196110

[21] "Raspberry pi 4 model b datasheet," [Online], Raspberry Pi Foundation, 2023, available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/.

[22] "Arduino mega 2560 rev3," [Online], Arduino, 2023, available: https://store.arduino.cc/products/arduino-mega-2560-rev3.

[23] "Lora/gps shield v1.4 user manual," [Online], Dragino, 2023, available: https://wiki.dragino.com.

[24] "Sx1276/77/78/79 lora transceivers – datasheet," Semtech Corporation, 2017.

[25] "Radiohead rh_rf95 lora library documentation," [Online], AirSpayce, available: www.airspayce.com/mikem/arduino/RadioHead.

[26] "L80 gps module hardware design (mtk3339 chipset)," [Online], Quectel Wireless Solutions, 2018, available: https://www.quectel.com.

[27] "Bots: An introduction for developers," [Online], Telegram, available: https://core.telegram.org/bots.