Enhancing Zigbee Communication Efficiency with Optimized FELACS Data Compression Algorithm in Python for IoT and WSN Networks

Raj Kumar, raj.kumar.596@my.csun.edu, Shahnam Mirzaei, shahnam.mirzaei@csun.edu, Member, IEEE

Abstract— **Efficient data handling in Zigbee-based Wireless Sensor Networks (WSNs) is crucial for the growing demands of Internet of Things (IoT) applications. This paper introduces FELACS, a data compression algorithm designed to enhance Zigbee communication efficiency. Implemented in Python, FELACS leverages a neighborhood indexing sequence to capture both spatial and temporal correlations in sensor data, achieving compression ratios of up to 75% for small packets. By reducing transmission time and energy consumption, the algorithm supports low-power, real-time communication in resource-constrained environments. Experimental results demonstrate that FELACS improves transmission efficiency while maintaining data integrity in applications such as environmental monitoring and healthcare. This work highlights FELACS as a promising solution for energy-efficient, bandwidth-optimized IoT systems, with potential for further developments in hybrid compression and security within Zigbee networks.**

Keywords—Zigbee, Compression, FELACS Algorithm, Wireless Sensor Network (WSNs), IoT, Adaptive Compression

I. Introduction

The Internet of Things (IoT) is driving a transformative technological revolution, connecting billions of devices to facilitate advanced automation, data collection, and analysis. Its applications span a wide range of industries, including smart cities, healthcare, agriculture, and industrial automation. In smart cities, IoT supports monitoring of traffic flow, energy consumption, and waste management. In healthcare, it enables remote patient monitoring by transmitting critical data to healthcare providers for timely intervention. In agriculture, IoT enhances resource management and boosts crop yield through precision farming, monitoring soil moisture, temperature, and crop health. Similarly, in industrial automation, IoT optimizes operations with predictive maintenance and improved efficiency.

However, despite its vast potential, IoT faces challenges due to the need for continuous data transmission in resource-constrained environments, which require efficient communication protocols. Zigbee, a low-power, low-data-rate wireless communication protocol, offers a promising solution for IoT, especially in battery-operated and remote applications. With its mesh network topology, Zigbee enhances reliability by enabling data routing through multiple nodes to reduce communication failures. Nevertheless, its limited bandwidth (up to 250 kbps) poses constraints in handling high data rates, real-time communication, and large data volumes, particularly in latency-sensitive applications like healthcare monitoring and industrial automation.

Zigbee's limited bandwidth and low data rate present significant challenges for real-time IoT applications that require high-frequency data transmission. These constraints can lead to issues such as data congestion, increased latency, and packet loss, undermining performance and reliability in mission-critical operations. While data compression presents a viable solution to reduce data size and transmission time, existing compression techniques need to be adapted to Zigbee's specific limitations. Effective solutions must strike a balance between compression efficiency, processing time, and energy consumption to ensure they are suitable for real-time applications.

The goal of this research is to design and evaluate a data compression framework specifically optimized for Zigbee networks, aiming to enhance bandwidth utilization, reduce transmission latency, and conserve energy. The key contributions of this work are as follows:

i. Develop a Compression Framework: Design a lightweight and efficient compression algorithm tailored to Zigbee networks to address the challenges posed by limited bandwidth.

ii. Improve Data Transmission Efficiency: Minimize data transmission time and optimize bandwidth usage to support real-time IoT applications.

iii. Enhance Energy Efficiency: Reduce power consumption in resource-constrained environments by integrating energy-efficient compression techniques.

iv. Evaluate Framework Performance: Assess the framework's effectiveness across various IoT application scenarios, including high-frequency data transmission and extended communication ranges.

This study focuses on the development and evaluation of a compression framework specifically for Zigbee networks, with particular emphasis on applications in healthcare, environmental monitoring, and industrial automation. The scope of the research excludes alternative communication protocols like Wi-Fi and LoRa to maintain a focused analysis on Zigbee. While security measures and advanced encryption are outside the scope of this study, they are recognized as potential areas for future exploration.

This research contributes to improving Zigbee network performance in IoT applications by addressing challenges related to bandwidth and energy efficiency through data compression. It demonstrates the potential of compression to facilitate efficient and reliable real-time communication, thereby extending device lifespans and supporting sustainable IoT solutions. Furthermore, the study provides valuable insights into the trade-offs between compression efficiency, latency, and energy consumption, setting the stage for future advancements in IoT communication technologies.

The structure of this paper is as follows: Section II discusses related work, Section III outlines the implementation of our proposed system, Section IV presents an analysis of the results, and Section V concludes the paper.

II. Related Work

WSNs and the IoTs have been pivotal in revolutionizing data-driven applications across domains such as environmental monitoring, healthcare, and industrial automation. Zigbee, a low-power and low-data-rate protocol, has gained traction for its energy efficiency and mesh networking capabilities. However, its limited bandwidth and real-time data transmission constraints have prompted the exploration of data compression techniques to enhance its efficiency.

Several studies have explored the integration of compression methods in Wireless Sensor Networks (WSNs). Zhang et al. [1] proposed a secure data-gathering scheme based on compressive sensing, which reduces data volume while maintaining security in resource-constrained environments. Gao et al. [2] demonstrated the potential of compressive sensing for next-generation WSNs, emphasizing bandwidth optimization. These approaches underscore the promise of advanced compression techniques to address the limitations of transmission capacity in Zigbee networks.

In addition to compression, researchers have developed adaptive algorithms designed specifically for IoT systems. Uthayakumar et al. [3] introduced Fast and Efficient Lossless Adaptive Compression Scheme (FELACS), which utilizes neighborhood indexing sequences to exploit spatial and temporal correlations in WSN data. This method achieved significant improvements in both compression ratios and energy efficiency. However, FELACS requires further optimization to ensure real-time responsiveness within Zigbee networks.

Energy efficiency continues to be a primary concern in WSNs. Ketshabetswe et al. [4, 10, 11] examined energy-aware routing protocols that minimize unnecessary transmissions, complementing the benefits of compression. Li et al. [5] applied denoising autoencoders to enhance data quality and reduce energy consumption in WSNs. These studies highlight the importance of balancing data integrity with processing overhead in Zigbee networks.

Despite these advances, there are still gaps in optimizing compression techniques specifically for Zigbee's unique characteristics. Few studies have explored the interaction between compression and Zigbee's low power and bandwidth limitations, leaving ample room for further research. Moreover, the integration of hybrid lossy and lossless compression methods, as suggested by Andrade et al. [6], has yet to be fully explored within the context of Zigbee networks.

Recent advancements in compression techniques also explore data reordering strategies to improve efficiency. Du [13] demonstrated that reordering sensor data prior to compression enhances redundancy patterns, leading to higher compression ratios and reduced transmission overhead. This method aligns well with FELACS' goal of optimizing Zigbee network performance, as it can further refine adaptive indexing and sequence-based compression techniques.

Gilbert et al. [14] proposed a trust-based data prediction and aggregation method using compressed sensing for clustered WSNs, reducing transmission overhead and enhancing data reliability. Similar techniques can complement our FELACS framework to optimize Zigbee network efficiency.

Maulidina et al. [15] conducted a comparative study on Zstandard, zlib, and LZ4, highlighting trade-offs in compression ratio, speed, and energy efficiency. Their findings support the selection of Zlib for higher compression and LZ4 for faster processing in resource-constrained environments. This research builds on the foundational work of FELACS and similar approaches by adapting compression schemes to meet Zigbee's constraints and simultaneously leveraging Zlib and Zstd. It seeks to fill the gaps in real-time data handling, energy efficiency, and bandwidth optimization while ensuring data integrity in critical IoT applications.

III. Implementation

This section outlines the proposed data compression framework for Zigbee-based WSNs. The framework aims to optimize bandwidth utilization, improve transmission efficiency, and preserve energy in resource-limited environments. It features a modular design with customized compression algorithms specifically for Zigbee, while also addressing system vulnerabilities to ensure reliable performance.

A. Bakground

The traditional FELACS algorithm, as described by the functional diagram depicted in Figure 1, was originally designed with a static compression model that does not adjust to the dynamic data patterns commonly seen in Zigbee networks. This results in a moderate compression ratio of around 50-60% for small data packets, which, while useful, is insufficient to fully optimize the limited bandwidth available in Zigbee networks. Moreover, the algorithm lacks integrated encryption, leaving the transmitted data vulnerable to potential security threats, which is a critical concern for modern IoT applications.

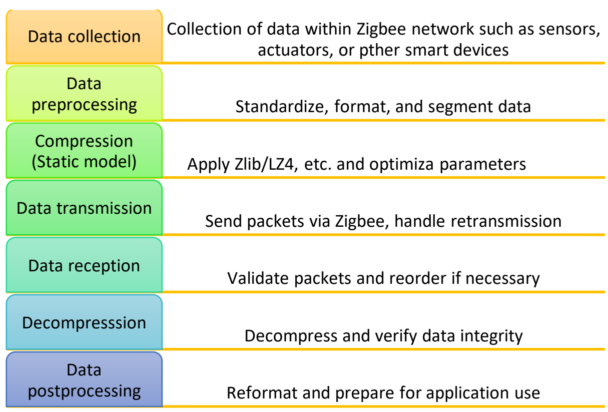


Fig. 1. Traditional FELACS algorithm

Another significant drawback is the relatively slow compression and decompression speeds, which introduce higher latency, particularly detrimental for real-time applications that require low-delay data transmission. In addition, the algorithm generates larger compressed data packets, which increases power consumption and consequently shortens the battery life of Zigbee devices. Given these inefficiencies in terms of compression, latency, and security, the traditional FELACS algorithm falls short of meeting the performance requirements of today's IoT ecosystems, where high efficiency, low latency, and robust security are essential. Although FELACS primarily focuses on lossless compression to maintain data integrity in Zigbee networks, recent research has explored lossy data compression strategies to achieve higher compression ratios [16]. Such approaches trade off data fidelity for reduced bandwidth consumption and energy efficiency, which could be valuable for non-critical sensor data transmission. Ebrahimi et al. [17] explored UAV-assisted data collection to reduce redundancy in WSNs using projection-based compressive sensing. While effective in mobile networks, our approach optimizes compression for static Zigbee networks to enhance communication efficiency. Prior studies [18] have examined the inherent challenges of ZigBee communication, including power limitations, interference, and restricted data rates, which our optimized FELACS algorithm aims to address.

The enhanced FELACS introduces substantial improvements for Zigbee-based WSNs to address key challenges like limited bandwidth, low data rates, and energy constraints. Notable advancements include dynamic neighborhood indexing for real-time data adaptability, achieving optimal compression ratios (up to 75% for small packets), and real-time parameter tuning to balance speed and efficiency for diverse IoT applications. Seamless integration with the Zigbee protocol ensures data integrity through packet segmentation, metadata tagging, and acknowledgment protocols, while energy efficiency is boosted by reduced transmission times and adaptive power systems, extending device lifetimes. Extensive testing confirmed its low latency, high transmission efficiency, and reliable data performance, with additional security features such as checksums and lightweight encryption. FELACS demonstrated strong applicability in real-world scenarios like healthcare, industrial automation, and environmental monitoring, enhancing network responsiveness and sustainability across various IoT applications. Existing studies, such as Hughes [19], analyze lossy and lossless compression impacts on time-series data, reinforcing the need for a tailored approach in Zigbee networks to balance efficiency and data fidelity.

Unlike the traditional FELACS algorithm, the proposed optimized FELACS algorithm, as illustrated in Figure 2, incorporates an adaptive compression model that adjusts in real-time to changing data patterns, leading to a significant improvement in compression efficiency. It achieves a higher compression ratio of up to 75% for small data packets, optimizing Zigbee's bandwidth usage. The proposed optimized FELACS algorithm also supports optional encryption, ensuring secure data transmission over Zigbee networks. Compression and decompression times are faster, resulting in lower latency, which is critical for real-time applications like healthcare monitoring and industrial automation. Additionally, the optimized FELACS algorithm reduces power consumption by minimizing data transmission size, extending the battery life of Zigbee devices. These enhancements make the optimized FELACS a more robust and scalable solution for modern Zigbee-based IoT systems.

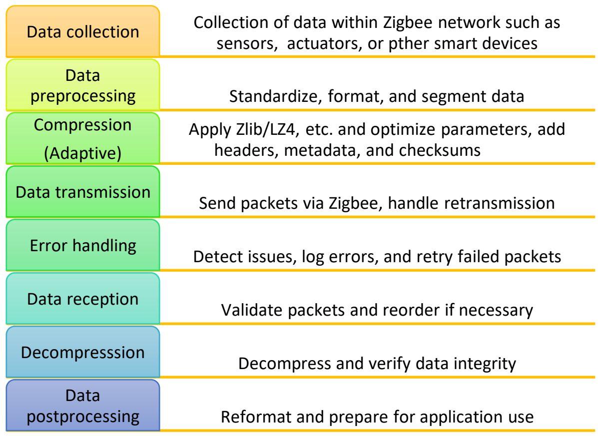


Fig. 2. Proposed optimized FELACS algorithm

B. System Architecture and Design Methodology

This paper leverages FELACS (Figure 1) as the core algorithm for optimizing data compression in Zigbee-based WSNs. The compression process begins with raw data input from Zigbee devices, followed by preprocessing to standardize the data format and segment larger data into packets that comply with Zigbee's size limitations. Depending on the application's needs, LZ4 [7] is used for high-speed compression, while open source Zlib [8] is chosen for its superior compression ratios. LZ4, known for its high-speed compression, has been widely used in consumer electronics for efficient data handling [20]. Its adaptability makes it suitable for Zigbee networks, optimizing bandwidth and energy consumption. The selected algorithm compresses the data and embeds essential metadata, such as sequence numbers and checksums, to ensure data integrity. The compressed packets are then encapsulated into Zigbee-compatible formats, queued for transmission, and sent to the receiver. This approach optimizes bandwidth usage and energy efficiency while maintaining reliable data transmission through integrity verification mechanisms.

The system utilizes a hardware setup shown in Figure 3 based on Zigbee modules (XBee Series 3) for efficient wireless communication in IoT applications, microcontroller or CPU (Raspberry Pi) handling data compression, transmission, and decompression tasks. Sensors act as data sources, providing input to the CPU, which preprocesses and compresses the data using algorithms such as FELACS, Zlib, or LZ4. The compressed data is prepared for transmission via universal asynchronous receiver transmitter (UART) serial communication and encapsulated into Zigbee-compatible packets. Zigbee modules then manage wireless transmission over a reliable mesh network to the receiving module, where the receiver CPU decompresses and processes the data for further use.

The system is optimized for low-power, resource-constrained environments, incorporating power management techniques like sleep modes for both transmitter and receiver CPUs and Zigbee modules, and supporting battery or solar-powered operations. We have used XBee configuration and test utility (XCTU) software to configure and test Zigbee modules, allowing adjustments to network parameters like personal area network identifier (PAN ID) and operating channels. Performance metrics, including compression time, transmission efficiency, and energy consumption, are evaluated through systematic testing with synthetic data to simulate real-world scenarios.

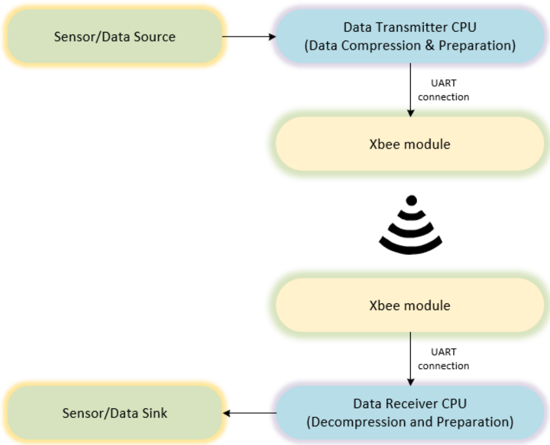


Fig. 3. Proposed system block diagram

The proposed practical workflow involves connecting hardware components via UART, deploying Python-based compression algorithms on CPUs, and validating system performance through testing and simulation. Development boards such as Raspberry Pi, Arduino, and ESP32 offer flexibility in implementing the framework across a range of IoT applications, including environmental monitoring, healthcare, and industrial automation. Additional tools like Wireshark and power profiling kits aid in debugging, optimization, and power efficiency analysis, ensuring a robust and scalable implementation. This framework provides a reliable and energy-efficient solution for wireless data communication in Zigbee-based networks.

Key software modules shown in Figure 4 include the Compression Module, which optimizes compression processes to reduce latency and enhance compression ratios; the Signal Simulation Module, which generates synthetic data to test the framework under varying load conditions; the Device Communication Module, which manages data transmission, acknowledgment protocols, and retransmissions for reliability; and the Network Management Module, which ensures stable data flow and error recovery. The Exception Handling Module strengthens network reliability by providing custom error protocols for retransmissions and diagnostics.

Integrating the framework with Zigbee networks involved adapting it to Zigbee's specific constraints, such as packet size limitations and network layer protocols. Challenges like compression overhead and data fragmentation were addressed through optimized parameter tuning and the use of metadata for sequence management. Testing across various transmission distances, data sizes, and compression times confirmed the framework's performance. Short-range tests demonstrated low power consumption and minimal interference, while medium- and long-range tests showed reliable transmission despite increased delays. Compression ratios reached up to 75%, resulting in significant improvements in bandwidth utilization, transmission efficiency, and energy consumption. Lightweight encryption and checksums ensured data security and integrity, making the framework highly suitable for real-time IoT applications, including healthcare monitoring, industrial automation, and environmental monitoring.

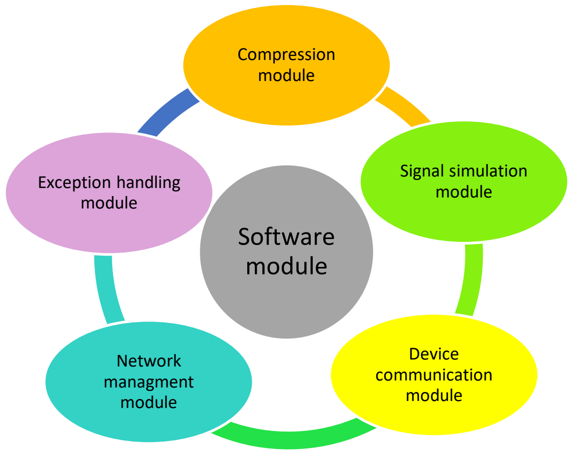


Fig. 4. Proposed adaptive FELACS software components

The proposed research work presented by this paper implements a modular, Python-based framework that optimizes the traditional FELACS algorithm to improve Zigbee network communication. An adaptive compression workflow depicted in Figure 5 was developed to ensure efficient data reduction by preprocessing data into standardized byte formats, segmenting large datasets, and utilizing Zlib or LZ4 algorithms based on specific application requirements. Compressed packets are tagged with metadata for integrity and reassembly, then formatted to meet Zigbee's packet size constraints and transmitted using acknowledgment protocols to guarantee delivery. The decompression workflow focuses on minimizing latency through packet verification, sequence reordering, and efficient decompression, restoring data to its original form for application use.

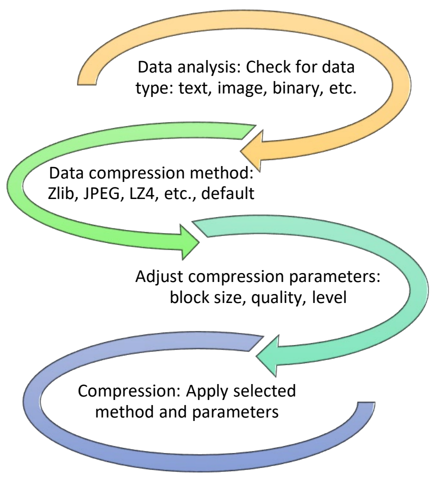


Fig. 5. Proposed adaptive compression process

The systematic testing methodology assessed the compression time and performance of FELACS in Zigbee-based WSNs. Compression times were categorized into short (under 1 ms), moderate (1~5 ms), and long (above 5 ms) durations based on data size and real-time application demands. Testing included unit tests for individual compression algorithms (e.g., Zlib, LZ4) across varying data sizes, real-world simulations with synthetic data, and precise timing measurements using Python's utilities. Key metrics evaluated included transmission efficiency (packet success rate, latency, bandwidth usage), energy consumption (power profiling for different data sizes and distances), and compression ratios (data size reduction percentage). The testing framework comprised unit tests for functionality, integration tests for end-to-end performance, and load tests to gauge resilience under varying data frequencies, payloads, and interference levels. These thorough evaluations confirmed FELACS's effectiveness for real-time, energy-efficient, and bandwidth-optimized IoT applications.

IV. Implementation Results

An optimized version of FELACS, specifically designed for Zigbee-based WSNs, was developed to incorporate an adaptive compression scheme. This enhanced FELACS effectively tackles the core challenges of Zigbee networks, including limited bandwidth, low data rates, and power constraints. By leveraging both spatial and temporal data correlations, the proposed algorithm dynamically adjusts compression parameters to optimize real-time data transmission and conserve energy, making it ideal for Zigbee's low-power IoT applications. Our results align with prior findings by Atasoy et al. [12], who demonstrated that selecting the appropriate compression algorithm can significantly enhance data transmission efficiency while maintaining computational feasibility. Similar trade-offs were observed in our FELACS implementation, where adaptive compression improved Zigbee network performance by reducing transmission delays and energy consumption.

Figure 6 illustrates the compression scheme integrated into the traditional FELACS algorithm. The compression was applied to 50 data samples of varying sizes. The compression framework exhibited an inverse relationship between packet size and compression ratio. Small packets (up to 256 bytes) achieved up to 80% compression, though the reduction was limited due to low redundancy. Medium packets (256 bytes to 1 KB) achieved a 65% compression ratio, benefiting from more redundancy in aggregated data, while large packets (greater than 1 KB) achieved 50% compression due to higher redundancy in complex datasets. These results highlight the framework's effectiveness for high-data-volume IoT applications, such as image or log transmission, where significant data reduction optimizes bandwidth and minimizes transmission time.

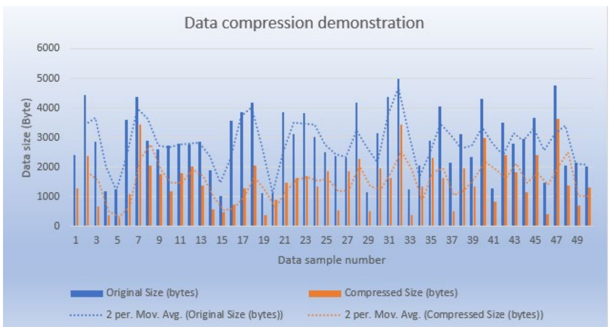


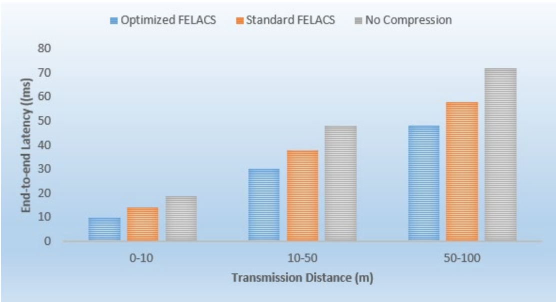
Fig. 6. Data compression demonstration

Transmission efficiency was assessed at short (0~10 m), medium (10~50 m), and long (50~100 m) distances. At short distances, transmission times were minimal (5~20 ms), with compression offering modest benefits. At medium distances, larger packets showed noticeable reductions in transmission delays due to compression. Long distances experienced the highest transmission times (up to 45 ms for large packets) due to signal attenuation and retransmissions, with compression reducing delays by 25%, thus enhancing reliability and speed for applications such as agricultural IoT or industrial monitoring. The framework's adaptability was tested across low (30 seconds), moderate (5~10 seconds), and high (1 second or less) data frequencies. Compression proved particularly beneficial in high-frequency scenarios by reducing retransmission rates and conserving power, which is crucial for battery-operated IoT devices. Power consumption decreased by 10~15% across all distances with compression, extending device lifetimes in energy-constrained environments.

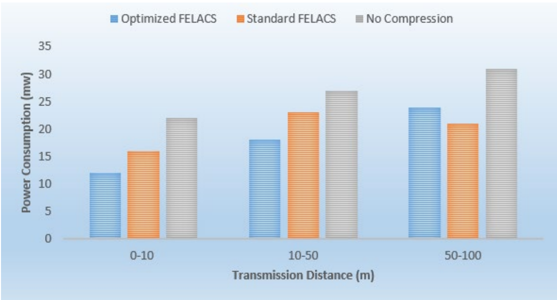
Compression significantly reduced end-to-end latency (Figure 7.a), especially for large packets over long distances, with a 25% improvement observed for 1 KB packets. The power consumption of a compression algorithm is closely tied to the transmission distance in wireless communication networks. Generally, the longer the transmission distance, the higher the power consumption, as the transmitter needs to increase its output power to maintain a reliable connection over greater distances. However, the use of a compression algorithm can help mitigate this by reducing the size of the data being transmitted, thus decreasing the number of transmission events required. This reduction in data size leads to fewer and shorter transmission bursts, ultimately conserving energy. Figure 7.b shows the power consumption of our proposed FELAC algorithm versus transmission distance for various data size packets. As the transmission distance increases, the benefits of compression become more pronounced, as it can reduce the overall transmission duration and frequency, offsetting the increased energy demand for longer-range communication. Therefore, while transmission distance directly impacts power usage, compression algorithms help optimize energy consumption by minimizing the data volume sent over the network. Compression and decompression times (Figure 7.c) remained minimal, with small packets requiring 2 ms and 1 ms, respectively, while large packets required 10 ms and 6 ms. These low latency times ensured the framework's suitability for real-time applications, such as healthcare monitoring and industrial automation.

Our proposed optimized FELACS outperformed existing algorithms such as Zlib and LZ4 by achieving higher compression ratios with lower latency. While Zlib provides good compression ratios, its complexity results in increased latency, and although LZ4 is faster, it struggles with larger datasets. Our proposed FELACS' dynamic parameter tuning allows it to adapt to a wide range of IoT scenarios, ensuring lossless data transmission and efficient handling of complex datasets.

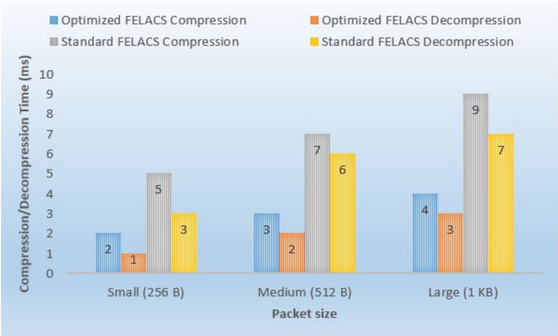
The framework's improvements in transmission efficiency, power savings, and latency reduction make it ideal for applications requiring real-time responsiveness, such as healthcare, industrial automation, and environmental monitoring. For instance, in healthcare, FELACS minimizes delays in transmitting patient data, ensuring timely monitoring and interventions. In industrial settings, it boosts operational efficiency by reducing delays in data-driven decision-making.



(a)



(b)



(c)

Fig. 7. End-to-end latency versus packet size measurements: (a) Power consumption versus transmission distance; (b) Transmission time versus transmission distance for different packet sizes; (c) Compression/Decompression time versus packet size

Despite its advantages, our proposed optimized FELACS has some limitations. Its adaptive nature introduces a slight processing overhead compared to simpler algorithms like LZ4, which may be challenging for ultra-low-power devices with limited computational resources. Additionally, integrating FELACS with Zigbee networks requires careful tuning of segmentation and error-handling protocols, adding to the implementation complexity. Furthermore, FELACS is specifically designed for Zigbee networks, making it less suitable for other IoT protocols, such as LoRa or Wi-Fi, without substantial modifications.

FELACS stands out due to its targeted optimization for Zigbee networks, which are characterized by low power consumption and low data rates. Unlike generic algorithms such as Zlib, LZ4, or Zstd [9], which are not optimized for Zigbee, FELACS adaptively adjusts compression parameters in real time to leverage data patterns, delivering superior performance in energy-constrained environments. Its seamless integration with Zigbee's network stack enhances compatibility, with features like custom segmentation, metadata embedding, and acknowledgment protocol capabilities that generic algorithms lack. Table 1 compares the features of the our proposed optimized FELACS algorithm against Zlib and LZ4 compression algorithms.

Table 1. Comparison of features for Optimized FELACS, FELACS, Zlib, and LZ4 compression algorithms

Feature	Optimized FELACS	Standard FELACS	Zlib	LZ4
Compression ratio	High	High	Very High	Moderate
Processing speed	Very High	High	Moderate	Very High
Energy efficiency	Very High	High	Moderate	High
Integration with Zigbee	Excellent	Excellent	Limited	Limited
Adaptability	Dynamic (Data Dependent)	Dynamic (Data Dependent)	Static	Static
Real time suitability	Excellent	Excellent	Moderate	Excellent
Implementation complexity	High	High	Low	Low

FELACS is a specialized solution for Zigbee-based WSNs, offering a balance of high compression ratios, real-time responsiveness, and energy efficiency. It surpasses generic algorithms like Zlib and LZ4 in Zigbee environments, particularly for applications that require real-time processing and energy conservation. Although its complexity and focus on Zigbee may limit its applicability to other wireless protocols, FELACS is an optimal choice for Zigbee networks that demand enhanced data compression and efficient communication.

V. Conclusion And Future Work

This paper presents the development and implementation of optimized FELACS, a custom solution designed for Zigbee-based WSNs. As the demand for efficient data transmission grows in resource-constrained IoT environments, and considering Zigbee's limitations in power and bandwidth, this innovation is particularly relevant. The proposed optimized FELACS algorithm tackles these challenges by achieving high compression ratios (up to 75% for small packets), minimizing data transmission, and supporting real-time decision-making for applications such as environmental monitoring and healthcare. Experimental results highlight optimized FELACS' ability to save energy, extend the lifespan of battery-powered devices, and reduce operational costs in remote or harsh environments, where periodic maintenance and battery replacement can be difficult. Overall, the proposed optimized FELACS algorithm proves to be a valuable solution for Zigbee networks, enhancing bandwidth efficiency, energy conservation, and real-time performance in dynamic IoT applications.

In summary, our proposed enhanced FELACS algorithm offers the following features:

1. Tailored Optimization for Zigbee Networks: Unlike general-purpose compression algorithms (e.g., Zlib, LZ4, Zstd), the proposed implementation specifically targets the constraints of Zigbee networks, such as low bandwidth, energy efficiency, and real-time performance.

2. Dynamic Adaptability: FELACS dynamically adjusts compression parameters based on real-time data patterns (spatial and temporal correlations), enabling efficient

performance across varying network conditions. Traditional algorithms like Zlib and LZ4 are static and do not adapt to changing data patterns.

3. Integration with Zigbee protocol stack: This implementation integrates compression and decompression workflows directly into Zigbee's protocol stack. It includes packet segmentation, metadata embedding (e.g., checksums and sequence numbers), and acknowledgment protocols, ensuring seamless data transmission with minimal overhead. Existing works often overlook such integration.

4. Error Handling and Resilience: The framework introduces robust error-handling mechanisms specific to Zigbee networks, such as handling simulated signal losses and transmission failures. This level of error management is not commonly addressed in other implementations.

Our study presented by our paper, has some limitations. First, the evaluation mainly focused on typical sensor data, which limits the understanding of how FELACS would handle more complex data types, such as images or multimedia, that may require alternative compression strategies. Second, the controlled experimental setting doesn't fully reflect real-world conditions like network congestion, environmental interference, or device mobility. Third, FELACS' computational demands may pose challenges for ultra-low-power devices with limited processing capacity, potentially leading to delays. Fourth, the scalability of FELACS in large-scale IoT deployments remains an area for further research, as increased data traffic and resource competition could affect network performance. Future research should focus on the items enumerated above. Furthermore, hybrid compression techniques that combine lossy and lossless methods to handle diverse data types, such as audio and video, where some data loss is acceptable for higher compression ratios. Finally, benchmarking against a broader range of modern compression algorithms beyond Zlib and LZ4 would provide deeper insights into FELACS' strengths and weaknesses, helping guide its deployment in diverse IoT ecosystems.

References

[1] Zhang, P., Wang, S., Guo, K., & Wang, J. (2018). A secure data collection scheme based on compressive sensing in wireless sensor networks. Ad Hoc Networks, pp. 70, 73–84.

[2] Gao, Z., Dai, L., Han, S., Chih-Lin, I., Wang, Z., & Hanzo, L. (2018). Compressive sensing techniques for next-generation wireless communications. IEEE Wireless Communications, 25(3), 144–153.

[3] Uthayakumar, J., Vengattaraman, T., & Dhavachelvan, P. (2019). A new lossless neighborhood indexing sequence (NIS) algorithm for data compression in wireless sensor networks. Ad Hoc Networks, 83, 149-157.

[4] Ketshabetswe, L. K., Zungeru, A. M., Lebekwe, C. K., & Mtengi, B. (2024). A compression- based routing strategy for energy saving in wireless sensor networks. Results in Engineering, 23, 102616.

[5] Li, G., Peng, S., Wang, C., Niu, J., & Yuan, Y. (2018). An energy-efficient data collection scheme using denoising autoencoder in wireless sensor networks. Tsinghua Science and Technology, 24(1), 86-96.

[6] Andrade, A., da Costa, C. A., Roehrs, A., Muchaluat Saade, D. C., & Righi, R. D. R. Blending Lossy and Lossless Data Compression Methods to Support Health Data Streaming in Smart Cities. Available at SSRN 4963549.

[7] Collet, Y., 2015. LZ4-Extremely fast compression. Technical Report.

[8] Jean-loup, G.A.I.L.L.Y., zlib: A Massively Spiffy Yet Delicately Unobtrusive Compression Library. http://www. zlib. net.

[9] Collet, Y. and Kucherawy, M., 2018. Zstandard Compression and the application/zstd Media Type (No. rfc8478).

[10] Ketshabetswe, K.L., Zungeru, A.M., Mtengi, B., Lebekwe, C.K. and Prabaharan, S.R.S., 2021. Data compression algorithms for wireless sensor networks: A review and comparison. IEEE Access, 9, pp.136872-136891.

[11] Ketshabetswe, L.K., Zungeru, A.M., Lebekwe, C.K. and Mtengi, B., 2024. Energy-efficient algorithms for lossless data compression schemes in wireless sensor networks. Scientific African, 23, p.e02008.

[12] Atasoy, F., Akkaya, A., Kocakır, N., & Karademir, Ö. (2023). Performance Analysis of Compression Algorithms on Matrix Data: Data Transfer Optimization in Microservices Architectures. Türk Doğa ve Fen Dergisi, (1), 49-60.

[13] Du, Q., 2024. Optimizing Data Compression via Data Reordering Strategies.

[14] Gilbert, E.P.K., Kaliaperumal, B., Rajsingh, E.B. and Lydia, M., 2018. Trust based data prediction, aggregation and reconstruction using compressed sensing for clustered wireless sensor networks. Computers & Electrical Engineering, 72, pp.894-909.

[15] Maulidina, A.P., Wijaya, R.A., Mazel, K. and Astriani, M.S., 2023, September. Comparative Study of Data Compression Algorithms: Zstandard, zlib & LZ4. In International Conference on Science, Engineering Management and Information Technology (pp. 394-406). Cham: Springer Nature Switzerland.

[16] Correa, J.D.A., Pinto, A.S.R. and Montez, C., 2022. Lossy data compression for iot sensors: A review. Internet of Things, 19, p.100516.

[17] Ebrahimi, D., Sharafeddine, S., Ho, P.H. and Assi, C., 2018. UAV-aided projection-based compressive data gathering in wireless sensor networks. IEEE Internet of Things Journal, 6(2), pp.1893-1905.

[18] Eren, H. and Fadzil, E., 2007, February. Technical challenges for wireless instrument networks-a case study with ZigBee. In 2007 IEEE Sensors Applications Symposium (pp. 1-6). IEEE.

[19] Hughes, J., 2023. Comparison of lossy and lossless compression algorithms for time series data in the Internet of Vehicles.

[20] Liu, W., Mei, F., Wang, C., O'Neill, M., & Swartzlander, E. E. (2018). Data compression device based on modified LZ4 algorithm. IEEE Transactions on Consumer Electronics, 64(1), 110-117.