RESEARCH ARTICLE OPEN ACCESS

M-SADR: Mobile Self Adaptive Data Rate for Enhanced LoRa End-Device Connectivity on the Move

Vasileios Moysiadis[1] | Panagiotis Radoglou-Grammatikis[2] | Evangelos Mourikis[2] | Maria Zevgara[2] | Dimitrios N. Skoutas[3] | Charalabos Skianis[3] | Panagiotis Sarigiannidis[1]

[1]Department of Electrical and Computer Engineering, University of Western Macedonia, Kozani, Greece | [2]Department of Research and Development, K3Y, Sofia, Bulgaria | [3]Department of Information and Communication Systems Engineering, University of the Aegean, Samos, Greece

Correspondence: Vasileios Moysiadis (bmoisiadis@uowm.gr)

Received: 27 November 2024 | **Revised:** 27 July 2025 | **Accepted:** 29 July 2025

Funding: This project has received funding from the European Union's Horizon Europe Research and Innovation Programme under grant agreement no. 101097122 (ACROSS).

Keywords: ADR | Localization | LoRa | LPWAN | Mobility | RSSI

ABSTRACT

Mobility is a critical aspect of the Internet of Things, reflecting the diverse and dynamic nature of potential applications. LoRa, a low power wide area network, is a promising wireless technology suitable for a wide range of applications. In this paper, we propose mobile self adaptive data rate (M-SADR), a method designed to handle LoRa-enabled mobile end devices, aiming to reduce energy consumption while maintaining a high packet delivery ratio. M-SADR operates on the end device side by selecting the spreading factor and transmission power for the next packet based on previous attempts. The proposed method stands out from similar solutions by enabling mobile LoRa end devices to determine transmission parameters efficiently, using a simple approach that requires minimal processing and storage resources. We evaluated our method in two distinct scenarios: the first involves three mobile gateways following a moving group of end devices, while the second uses a single static gateway with multiple mobile end devices moving arbitrarily around it. The evaluation of the two simulation scenarios showed that our method achieves better results, with up to 2.4% higher packet delivery ratio and 5.7% lower energy consumption per successfully delivered packet compared with other methods (ADR-MIN, BADR, and HADR) for LoRa mobile end devices. Furthermore, a small-scale testbed was implemented, demonstrating that the proposed algorithm performs better than its competitors in real-world scenarios. Finally, in the first simulation scenario, we present a methodology uses RSSI-based position estimation to reposition the mobile gateways and maintain proximity to the moving group of end devices.

1 | Introduction

Mobility is one of the main characteristics of end devices in the Internet of Things (IoT). A high percentage of end devices operate autonomously without human intervention and need to be connected to send or receive data. Currently, numerous IoT applications rely on or incorporate mobility. Examples include smart cities, connected vehicles, bike rental systems, smart healthcare, aging population support, fleet monitoring, and wildlife or animal tracking [1]. In more detail, mobility is a core component of smart city solutions. Connected vehicles, smart traffic lights, and public transportation systems leverage IoT mobility features to reduce congestion, improve safety, and enhance urban planning [2]. Furthermore, mobility enables IoT devices to gather and transmit data in real-time, facilitating timely decision-making. For instance, in fleet management, IoT devices continuously update vehicle locations, allowing companies to make more informed decisions and improve logistics efficiency [3].

In parallel with research focused on mobile end devices, considerable attention has been given to sink mobility in wireless sensor networks, where mobile sinks (or gateways) are used to collect data from static or semi-static sensor nodes. This approach aims to reduce energy consumption and balance communication loads by minimizing the number of multi-hop transmissions required from distant nodes [4, 5]. Notable protocols, such as E2SR2 [6], Ring Routing [7], and Railroad Routing [8], have been proposed to optimize data gathering through planned or adaptive sink movements. While these protocols have demonstrated significant improvements in network performance, they address a fundamentally different set of challenges compared with scenarios involving mobile end devices, which are inherently more unpredictable and decentralized in their behavior.

Beyond ground-based mobility, unmanned aerial vehicles (UAVs) can play a significant role in the IoT due to their versatility and ability to collect and transmit data from otherwise inaccessible or challenging environments. For example, UAVs can collect data from remote or hard-to-reach areas, such as disaster zones, agricultural fields, and construction sites [9, 10]. In addition, Self Organised Networks (SONs) have gained significant importance in recent years, playing a crucial role in various IoT applications and aiming to transform multiple technological areas. Some of the main SON models include improving mobility, balancing network traffic, reducing interference, and optimizing capacity and coverage [11].

As IoT ecosystems expand, mobility will play a key role in scaling solutions across smart cities, industrial automation, and emerging technologies such as autonomous drones and robotic delivery systems. IoT devices must remain connected regardless of movement. Technologies like 5G, low power wide area networks (LPWANs), and edge computing enhance connectivity, ensuring seamless data transmission even in mobile environments.

Additionally, reducing energy consumption is another crucial factor for autonomous end devices, particularly when using LPWANs like LoRa, where minimizing energy consumption is a significant advantage. LoRa's long-range capabilities and low-power design are well-suited for mobile applications, but optimizing these features for dynamic environments is essential. Efficient energy management and advanced protocols for mobile LoRa devices can significantly enhance performance and extend battery life, ensuring reliable connectivity while maintaining low energy consumption.

Research on LoRa (long range) technology has primarily focused on stationary applications, leaving a notable gap in the optimization of mobile end devices. As self-organized networks become more prevalent, optimizing LoRa for mobility is crucial for reducing energy consumption. Advances in low-power design, efficient communication protocols, and scalable network management are essential for enhancing the battery life and performance of mobile LoRa devices. Understanding how mobility affects connectivity can lead to more energy-efficient solutions while maintaining reliable operation.

In this work, we focus on a self-organized network of mobile LoRa end devices supported by mobile LoRa gateways mounted on UAVs. The aim of our work is two-fold. First, we introduce the mobile self adaptive data rate (M-SADR), an alternative adaptive data rate (ADR) method adapted for mobile end devices. Secondly, we propose a methodology for repositioning the LoRa gateways to ensure they remain in close proximity to a moving group of mobile LoRa end devices.

More specifically, the proposed methodology for supporting mobile LoRa end devices with mobile gateways is an improved version of our previous work [12]. The mobile LoRa end devices move arbitrarily as a group within a large area, while mobile gateways, adapted to UAVs, follow the movement of the group of end devices. The actual position of the end devices are directly known but are estimated based on RSSI values of their transmitted signals. In this work, we have improved our previous methodology to support end device movement up to 24 mps (meters per second). Moreover, gateways must move at twice the speed of the moving group of nodes to maintain proximity to the end devices. This constitutes a significant improvement compared with our previous work.

The proposed methodology can be valuable in various applications, such as disaster scenarios or situations where autonomous cooperative mobility, is crucial. In many disaster scenarios, supporting operational teams in the field of operation is critical to ensuring effective response and coordination. Knowing the exact location of each person or piece of equipment enables the leadership team to efficiently coordinate available resources during the critical first hours of a disaster. Moreover, there is significant interest in developing disaster management systems that can save lives, protect property, and minimize the substantial economic costs associated with such events. In this context, LoRa technology has emerged as a promising solution [13].

In addition, the proposed M-SADR algorithm performs better than existing approaches for mobile LoRa end devices. We compare its performance against three alternative methods suggested for mobile end devices, namely BADR [14], HADR [15], and ADR-MIN [16]. The results demonstrate that M-SADR achieves a high packet delivery ratio (PDR), even when the network traffic is considerably high. This is achieved by efficiently allocating the LoRa parameters, resulting in lower energy consumption per successfully transmitted packet. More specifically, M-SADR consistently demonstrates lower energy consumption than competing methods across all tested speeds and network traffic conditions.

Moreover, the proposed M-SADR approach has the advantage of retaining recent transmission conditions in the terminal device's memory through a simplified algorithm with minimal processing power and storage requirements. This allows the terminal device to independently decide on transmission parameters without consuming significant processing power or energy, which is crucial for LoRa devices with energy constraints.

In contrast, the compared algorithms, BADR and HADR, also employ simple, low-requirement algorithms, but they send packets without considering current conditions, except for HADR, which depends on commands from the network server when the

TABLE 1 | Comparison of methods for efficient data rate management.

	Mobile	Network server	End device	Machine learning	Features
ADR	—	✓	✓	—	SF, TP
ADR+ [26]	—	✓	✓	—	SF, TP
ADR-MIN [16]	—	✓	✓	—	SF, TP
DDQN-PER [27]	—	✓	✓	✓	SF, TP
BADR [14]	✓	—	✓	—	SF, TP
HADR [15]	✓	✓	✓	—	SF, TP
LR±ADR [28]	✓	✓	✓	✓	SF, TP
G-ADR [29]	✓	✓	✓	—	SF, TP
EMA-ADR [29]	✓	✓	✓	—	SF, TP
CA-ADR [30]	✓	✓	✓	✓	SF
RADR [31]	✓	✓	✓	✓	SF
PST [32]	—	✓	✓	✓	Scheduling
C-ADR [33]	—	✓	✓	✓	SF, Channel
ASA [34]	—	✓	✓	✓	Scheduling
EARN [35]	—	✓	—	—	SF, TP, CR
ADR-CC [36]	—	✓	✓	✓	SF, backoff
E-ADR [37]	✓	✓	✓	—	SF, CR, BW

terminal device is designated as static at a particular moment. Additionally, the ADR-MIN algorithm continuously relies on commands from the network server, making it less effective in networks with high traffic.

The contribution of our work is summarized as follows:

- We propose a new M-SADR algorithm that consumes less energy per successfully delivered packet.
- We compare the proposed M-SADR method with competitor algorithms for mobile end devices, demonstrating advantages in PDR and energy consumption, particularly under high network traffic conditions.
- We improved our previous methodology for autonomous mobile gateways to better support mobile end devices. The mobile gateways can now support end devices moving at speeds up to 24 mps. In addition, they must move at no more than twice the speed of the end devices to maintain proximity.

The rest of this paper is organized as follows: In Section 2, we discuss the key features of LoRa and the theoretical background of localization techniques in wireless sensor networks. Additionally, we present the most relevant research efforts on mobile LoRa end devices and localization methods. In Section 3, we provide an in-depth analysis of the proposed M-SADR algorithm and present the detailed methodology for repositioning mobile LoRa gateways to maintain proximity to mobile end devices. We also outline the main parameters used in the two simulation scenarios. Next, in Section 4, we present an evaluation of our proposed M-SADR mechanism compared with three other alternatives: ADR-MIN, BADR, and HADR, designed to support mobility in LoRa networks. Section 6 discusses the evaluation results, highlighting the advantages and acknowledging the potential limitations of the proposed method. Finally, Section 7 concludes this paper.

2 | Background

In this section, we present the main features of LoRa and provide a brief review of ADR methods and similar approaches for LoRa end devices. Furthermore, we discuss the most common localization techniques used with LoRa.

2.1 | LoRa

LoRa belongs to the category of LPWANs, which is one of the main categories of wireless networks utilized in the IoT. LPWANs are often used in IoT applications not only in urban but also in rural areas, where they excel over alternative solutions. The main features of this category are the low energy consumption and the long-distance transmission. However, the main disadvantage is the low data rate. Despite this, LPWANs are widely used in many IoT applications, such as Smart Farming [17, 18], Smart Cities [19], Smart Metering [20], and Industry 4.0 [21] applications.

LoRa is one of the most popular wireless networks in LPWAN category, offering several advantages over its competitors, such as SigFox and NB-IoT. More specifically, it uses a free license band with no licensing fee, is easy to deploy, provides an acceptable data rate, and consumes low energy. Additionally, end device costs are relatively low, and no additional hardware is needed if a LoRa network is already deployed in the area. It operates in the unlicensed band of 868 MHz in Europe, 915 MHz in America, and 430 MHz in Asia. LoRa can reach distances up to 20 km with a maximum data rate of 50 kbps per channel. It operates with a duty cycle of 1% to ensure fairness between devices, enabling support for numerous end devices that work simultaneously in the same area. This makes it suitable for a wide range of applications. LoRa uses Chirp Spread Spectrum (CSS) modulation, which facilitates the transmission of messages over long distances.

Despite its technical advantages and growing adoption in various sectors, LoRa-based applications also face important challenges, particularly in the area of security. Due to their open, resource-constrained, and decentralized nature, these systems face significant threats, including Sybil attacks, wormhole attacks, jamming, and selective forwarding, all of which compromise data integrity, availability, and confidentiality. These vulnerabilities are particularly severe in LoRa systems, where long-range, low-power communication increases the attack surface while limiting the ability to implement strong cryptographic defenses. The authors in [22, 23] provide comprehensive surveys of such attacks in WSNs, highlighting the need for lightweight, adaptive countermeasures tailored to constrained IoT environments like LoRaWAN.

2.1.1 | LoRa Main Parameters

The main parameters of LoRa are the spreading factor (SF) and the transmission power (TP). More specifically, the SF parameter defines the time required for a symbol to transmit over the air. It typically ranges from SF7 to SF12, or from SF6 to SF12 in newer chipsets. A message sent with SF7 takes less time to transmit and consumes less energy but cannot reach long distances. In contrast, a message sent with SF12 requires significantly more transmission time and energy but can reach much longer distances.

The TP parameter defines how much energy is used to transmit a signal. Higher TP values consume more energy and enable transmission over longer distances, while lower values conserve energy but limit the transmission range to shorter distances. Most devices support TP values up to 20 dBm, although the maximum allowed is subject to local regulations. For example, in Europe, the maximum allowed limit is restricted to 14 dBm.

Another parameter defined by LoRa is the coding rate (CR), which determines the additional bits included in the packet for error detection. For example, the default value is 4/5, meaning that one extra bit is added for every four useful bits. Including more extra bits enhance error detection, especially in noisy environments, but also increases energy consumption.

LoRa also defines three different values for the channel bandwidth (BW): 125, 250, and 500 kHz. The BW used depends on local regulations. For example, in Europe, only BW125 and BW250 are used. Additionally, BW and SF jointly define the data rate of packet transmission. For instance, BW250 provides double the data rate of BW125 for the same SF.

Furthermore, SF and BW determine the time on air (ToA) required to transmit a packet over a channel with a given bandwidth and SF. A higher SF leads to a longer ToA, which can be a significant drawback, especially in high-traffic scenarios. The ToA calculation is thoroughly explained in [24].

Moreover, LoRaWAN defines three different classes for end devices: Class A, Class B, and Class C [25]. Class A devices are designed for those with restricted energy consumption capabilities. They are usually in deep sleep mode and wake up only to transmit messages, while they can only receive messages after a successful transmission. In contrast, Class B devices can receive

ALGORITHM 1 | M-SADR algorithm on end device.

Algorithm 1 M-SADR algorithm on end device

```
Input: $P$                  ▷ Array with probabilities of all SFs (SF7-SF12)
Output: $SF$, $TP$                                      ▷ Selected $SF$, $TP$
1: Procedure: ReceivingAcknowledgement
2: $cntACK \Leftarrow$ Update from last 10 packets            ▷ counter for ACKs
3: $b \Leftarrow 0.05 * (10 - cntACK)$
4: if $b < 0.05$ then
5:     $b \Leftarrow 0.05$
6: end if
7: if $ACK$ received then
8:     $R_{ACK} \Leftarrow 1$
9: else
10:    $R_{ACK} \Leftarrow 0$
11: end if
12: $P[SF] \Leftarrow (1 - b) \cdot P[SF] + (b \cdot R_{ACK})$
13:
14: Procedure: SendingPacket
15: $SF, P_{max} \Leftarrow max(P)$                      ▷ Find the maximum of $P$ and
16:                                                ▷ select the corresponding $SF$
17: if $SF$ equal to previous $SF$ then
18:     if two consecutive successfully delivered packets then
19:         if $TP > 10$ then                              ▷ Decrease TP if possible
20:             $TP \Leftarrow TP - 2$
21:         else                                           ▷ Decrease SF if possible
22:             if $SF > 7$ then
23:                 $P[SF - 1] \Leftarrow P[SF]$
24:                 $SF \Leftarrow SF - 1$
25:                 $TP \Leftarrow 14$
26:             end if
27:         end if
28:     end if
29:     if two consecutive unsuccessfully delivered packets then
30:         if $TP < 14$ then                              ▷ Increase $TP$ if possible
31:             $TP \Leftarrow TP + 2$
32:         else
33:             $R_{ACK} \Leftarrow 1$
34:             for $SF \neq$ current $SF$ do
35:                 $P[SF] \Leftarrow (1 - b) \cdot P[SF] + (b \cdot R_{ACK})$
36:             end for
37:         end if
38:     end if
39: else                                                ▷ We have new selected SF
40:     $TP \Leftarrow 14$
41: end if
42: Send packet with $SF$, $TP$
```

or transmit messages at any time, but only within specific time slots, requiring a mechanism to ensure synchronization between all end devices and gateways. Finally, Class C devices can transmit messages at any time and receive messages only when they are not transmitting.

2.1.2 | Efficient Data Rate Management

LoRaWAN includes the ADR mechanism, which optimizes transmission parameters to enhance efficiency. ADR dynamically adjusts the SF and TP to balance energy consumption and ensure reliable data delivery. Designed for stationary devices, the default ADR mechanism is divided into two parts: one running on the end device and the other on the network server.

The network server part of the default ADR algorithm calculates the maximum signal-to-noise ratio (SNR) from the last 20 received packets, which is then used to estimate the optimal parameters for SF and TP. In practice, the network server only decreases SF and TP to reduce energy consumption while maintaining reliable data delivery. On the end device side, the mechanism follows a simple approach, increasing SF and TP only after consecutive packet losses to regain connectivity.

Many alternatives to ADR have been proposed for static end devices. One such example is ADR+ [26], which calculates the average SNR from the last 20 packets. This method provides a more stable estimate of signal quality by equally considering all SNR values rather than focusing on just one maximum value. While this approach helps maintain connection reliability, it may not be as sensitive to changes in signal quality compared with other dynamic methods.

ADR-MIN [16] is another one alternative based on the default ADR algorithm, but calculates the minimum SNR from the last 20 packets, instead of the maximum value used in the standard ADR. This approach can be more conservative, adjusting parameters based on the worst network conditions. While ADR-MIN achieves excellent results in terms of PDR, it often results in excessive energy consumption, even when

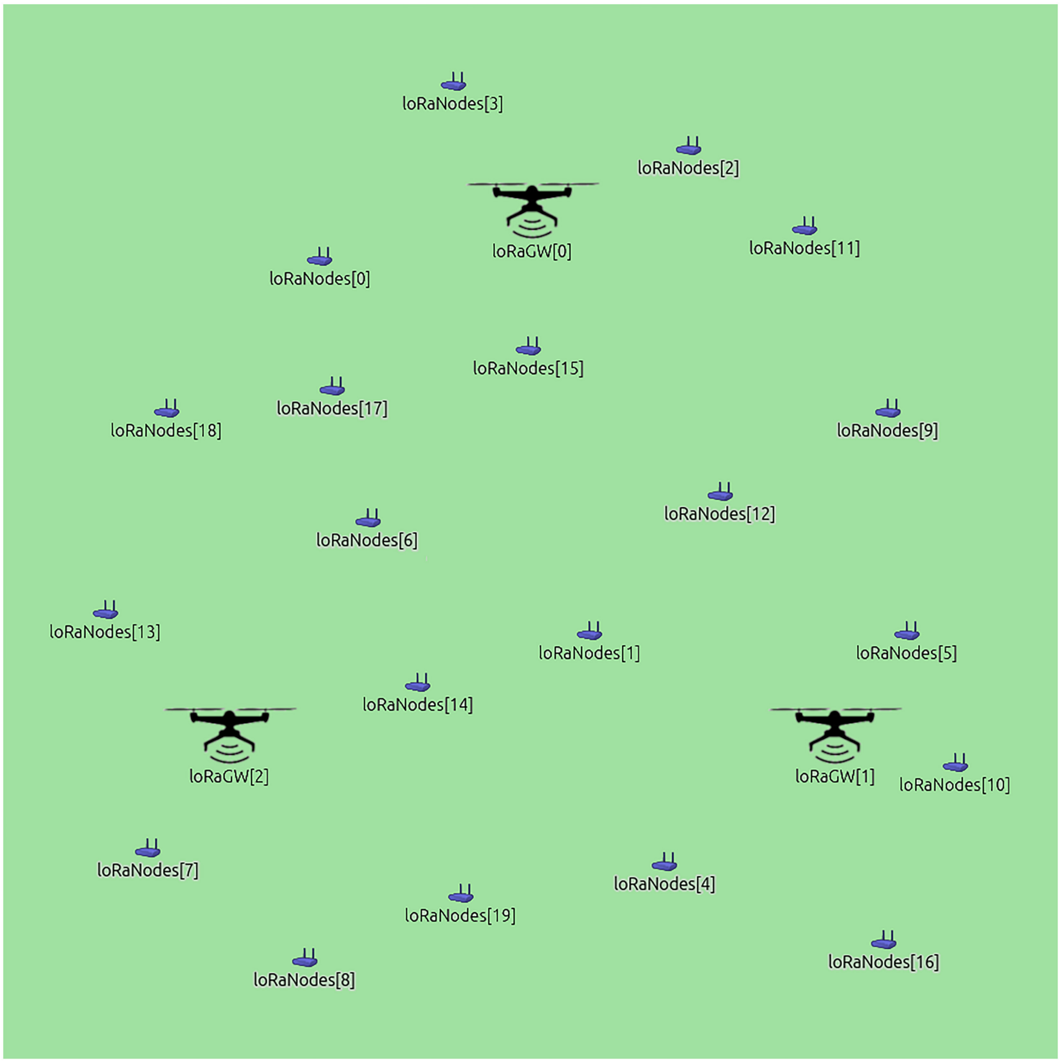


FIGURE 1 | Close-up of the moving group of nodes with the three gateways in the simulation area.

TABLE 2 | Simulation parameters.

Number of nodes	20, 100, 200, 300, 400
Number of gateways	3
Simulation area	50 km × 50 km
Node altitude	0 m
Gateway altitude	150 m
Node speed	(1, 8, 16, 24) mps
Gateway speed	(2, 16, 32, 48) mps
Initial SF	12 or algorithm depended
Initial TP	14 or algorithm depended
Number of retries	1
Node mobility	SuperpositioningMobility AttachedMobility RandomWaypointMobility
Group wait time	0–1200 s
Node wait time	0–600 s
Time for first packet	uniform (0 s, 120 s)
Time for next packet	60 s + exponential (60 s)
Path loss model	ShadowLogNormal
$d0$	400
γ	2.08
σ	3.57

not necessary, due to its conservative nature. This can lead to inefficient resource usage, as the device may continue using higher SF and TP values even when the connection does not require such strict parameters.

In [27], the authors present the DDQN-PER reinforcement learning algorithm for optimizing network transmission parameters, specifically SF and TP. The algorithm effectively balances energy consumption and PDR by prioritizing critical transmissions and accelerating optimal parameter selection. It consistently outperforms ADR and RL-based methods, particularly in large-scale networks with 1000 devices, maintaining superior performance in both obstacle-laden and open environments.

Regarding mobile end devices, Semtech proposes the blind adaptive data rate (BADR) [14], a method designed to work efficiently than default ADR. BADR operates by repeatedly transmitting packets using only SF7, SF10, and SF12, without requiring feedback from the network server. This approach is particularly beneficial in mobile scenarios, where frequent changes in signal strength and interference occur.

Similarly, the authors in [15] introduce the hybrid adaptive data rate (HADR), which is suitable for both mobile and static end devices, as it combines ADR+ and BADR. Specifically, when the network server detects that the device is static, the ADR+ mechanism is activated. Otherwise, the end device transmits packets sequentially, using SF7 through SF12. The main drawback of HADR is that it requires additional bytes in the packet header to transmit the position of the end device.

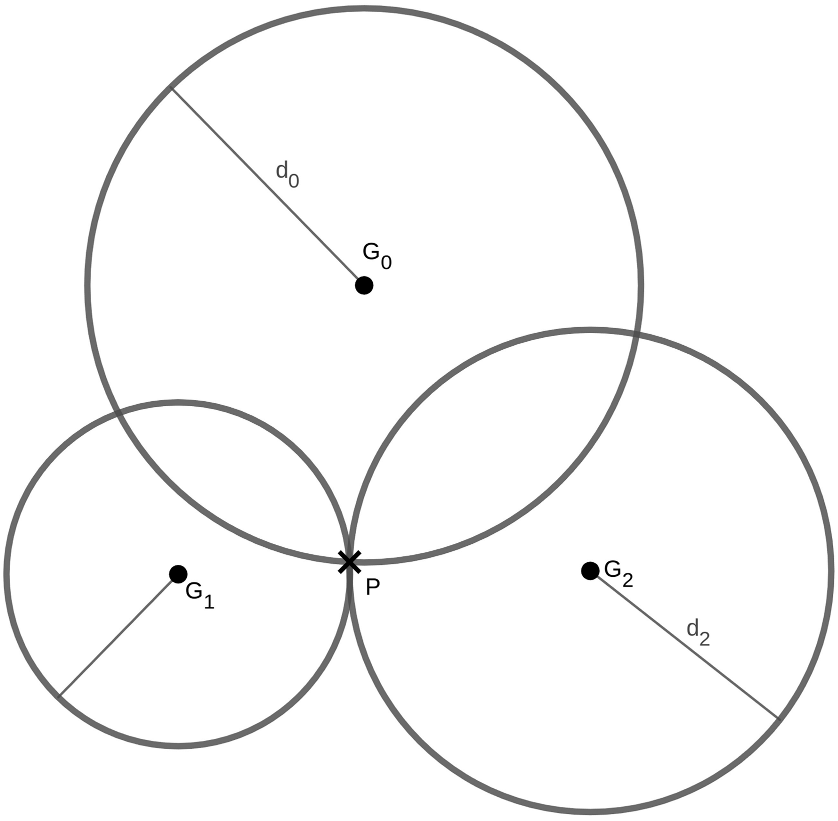


FIGURE 2 | Three gateways received the packet and all circles intersect in one common point.

Moreover, in our previous work [28], we proposed LR-ADR and LR+ADR, which are designed for mobile end devices with static LoRa gateways. These ADR methods leverage linear regression to predict the next expected SNR, enabling the calculation of optimal SF and TP parameters for the node.

The authors in [29] propose G-ADR and EMA-ADR, two mechanisms designed to optimally assign SF and TP for both static and mobile end devices. These methods aim to reduce the convergence period of end devices, thereby enhancing data transmission efficiency and improving energy management by enabling faster adaptation to changing network conditions.

The authors in [30] present the contextual aware ADR (CA-ADR) to improve LoRaWAN's ADR mechanism through a two-phase approach. In offline mode, it compiles and refines a dataset using contextual rule-based learning (CRL) before training a hybrid CNN-LSTM model. In online mode, the pre-trained model is deployed for efficient resource allocation to both static and mobile end devices. Implemented using TinyML for low-power devices, CA-ADR improves packet success ratio (PSR) and energy efficiency, making it a promising solution for IoT applications.

The authors in [31] propose an AI-driven LoRaWAN system with the robust ADR (RADR) transmission strategy for mobile location-based services. RADR initially selects the highest data rate (i.e., the lowest SF) for energy efficiency, then employs an AI model to determine the lowest stable data rate (i.e., the highest SF) based on uplink history, ensuring reliable packet delivery. This approach provides a more stable mobile IoT service without excessively lowering the data rate, making the AI-driven RADR effective for mobile applications.

Furthermore, methods not related to the ADR mechanism have been proposed to increase network efficiency and reduce energy consumption. For example, the authors in [32] introduce a dynamic transmission priority scheduling technique (PST) utilizing an unsupervised learning clustering algorithm to minimize packet collisions and improve network transmission delay and energy efficiency. Specifically, the LoRa gateway categorizes nodes into distinct transmission priority clusters. The dynamic PST enables the gateway to adjust transmission intervals based on the priority levels of these clusters. This approach supports network scalability while maintaining a low packet collision rate (PCR) and significantly enhancing transmission delay and energy efficiency.

The authors in [33] introduce cluster ADR (C-ADR), a mechanism designed to reduce collisions and packet loss in dense LoRa-based Industrial IoT networks. By leveraging signal orthogonality, C-ADR strategically allocates SFs and clusters end devices based on estimated path loss. The approach assigns different channels to each cluster and optimally distributes SFs among devices to enhance network performance. The effectiveness of C-ADR is validated through simulations and testbed implementation, assessing packet success rate, power consumption, and convergence time.

In [34], the authors focus on optimizing delay in LoRaWAN through an adaptive scheduling algorithm (ASA) combined with an unsupervised probabilistic approach, the Gaussian

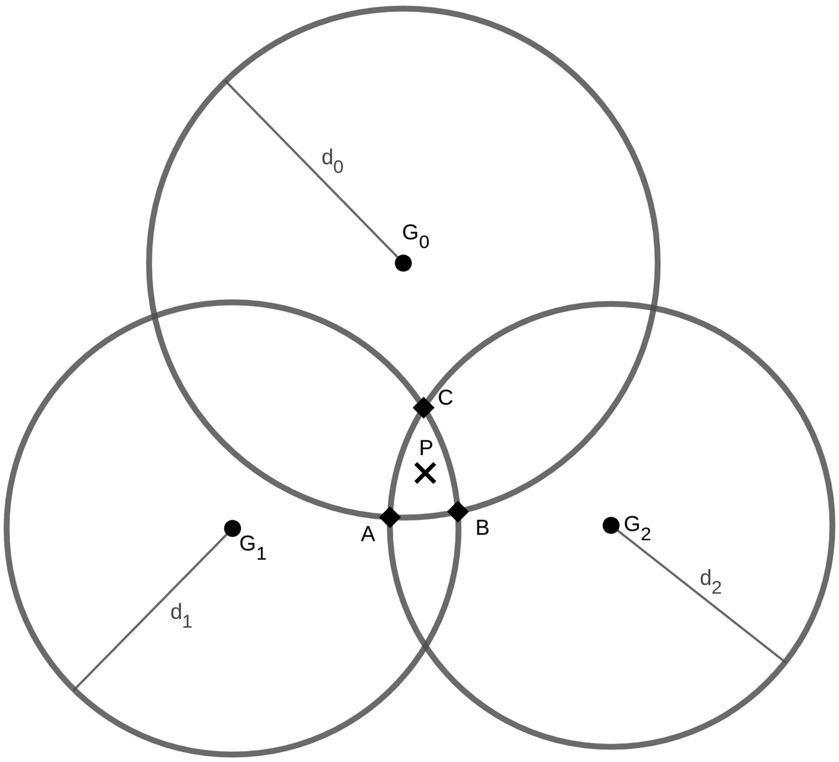


FIGURE 3 | Three gateways received the packet and all circles intersect forming a common area.

mixture model (GMM). This approach minimizes retransmissions, leading to improved delay performance in LoRaWAN. The results demonstrate that their method significantly reduces the PCR compared with conventional LoRaWAN. Additionally, the PSR is notably increased compared with both conventional LoRaWAN and the dynamic PST.

The authors in [35] propose an enhanced ADR (EARN), a server-side mechanism that extends the standard ADR by jointly optimizing the SF, TP, and CR. Unlike traditional ADR schemes, EARN introduces a performance-driven selection strategy based on analytical models that estimate collision probability, frame error rate, and energy-per-packet. It operates entirely at the network server without requiring any additional computation at the end-device side, preserving the low-power operation of constrained nodes. It uses deterministic models and exhaustive evaluation of parameter combinations to find the most energy-efficient configuration. Simulation results show that EARN significantly improves energy efficiency and delivery reliability, especially in dense deployments with fluctuating link conditions.

The authors in [36] propose an adaptive data rate with congestion classifier (ADR-CC), a machine learning-based ADR scheme that enhances network performance under congestion. The approach employs a logistic regression classifier trained at the network server, using features such as data rate, RSSI, number of connected devices, and acknowledgment reception ratio. The trained model is then transmitted to end devices, which perform local inference to determine the presence of congestion and accordingly adjust either the data rate or the backoff time. ADR-CC combines centralized learning with lightweight device-side execution, offering adaptability with minimal computational overhead. The method targets static deployments and does not address mobility scenarios.

The work presented in [37] proposes Enhanced ADR (E-ADR), a rule-based mechanism that extends the standard ADR to support mobile end-devices. By estimating the device's position using trilateration and RSSI-based displacement models, the network server dynamically adjusts key transmission parameters, such as SF, CR, and BW, in order to reduce energy consumption and packet loss. Unlike many recent approaches, E-ADR does not rely on machine learning techniques, which keeps complexity low and enables real-time operation. Experimental results in mobile smart farming scenarios demonstrate significant improvements in time-on-air, energy efficiency, and reliability compared with the baseline ADR scheme.

Table 1 summarizes the main features of the methods discussed earlier. More specifically, it indicates whether each method was tested with mobile end devices and whether it employs a machine learning approach. It also specifies whether components of the method are executed on the network server and/or the end device. Finally, the last column highlights the specific parameters that are adjusted to enhance performance.

In addition, there are some research efforts on mobile LoRa end devices that are application specific. For example, the authors in [38] propose an improved ADR for agricultural mobile sensors.

Furthermore, other approaches for collecting data from LoRa devices have been proposed. The authors in [39] propose MLoRaDrone, which uses multiple UAVs equipped with LoRa gateways flying near nodes. Their work aims to minimize the energy consumption of end devices by shortening transmission distances. In [40], the authors introduce an innovative architecture to ensure reliable data collection in pristine environmental sites, utilizing a mobile, vehicle-mounted LoRaWAN gateway.

2.2 | Localization

The most common localization method is utilizing the Global Navigation Satellite System (GNSS), which includes four different constellations: Global Positioning System (GPS) (USA), Galileo (Europe), GLONASS (Russia), and BeiDou (China). All

ALGORITHM 2 | Estimate node position with three gateways.

Algorithm 2 Estimate node position with three gateways

```
Input: $G_0(x_0, y_0), G_1(x_1, y_1), G_2(x_2, y_2)$    ▷ All GWs received the packet
Input: $d_0, d_1, d_2$ with $(d_0 > d_1 > d_2)$           ▷ Distances sorted ascending
Output: $(P_x, P_y)$                                      ▷ Estimated position of node
1: $d_{01} \Leftarrow \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}$            ▷ Distance $G_0, G_1$
2: $d_0' \Leftarrow d_0$
3: $d_1' \Leftarrow d_1$
4: if $d_{01} > (d_0 + d_1)$ then
5:     $temp \Leftarrow d_{01} - (d_0 + d_1)$
6:     $d_0' \Leftarrow d_0' + temp/2$
7:     $d_1' \Leftarrow d_1' + temp/2$
8: else
9:     if $d_{01} < abs(d_0 - d_1)$ then
10:        $temp \Leftarrow abs(d_0 - d_1) - d_{01}$
11:        if $d_0 < d_1$ then
12:            $d_0' \Leftarrow d_0' + temp/2$
13:            $d_1' \Leftarrow d_1' - temp/2$
14:        else
15:            $d_0' \Leftarrow d_0' - temp/2$
16:            $d_1' \Leftarrow d_1' + temp/2$
17:        end if
18:    end if
19: end if
20: $l \Leftarrow (d_0' - d_1' + d_1^2)/(2d_{01})$
21: $b \Leftarrow \sqrt{(d_1^2 - l)^2}$
22: $x_a \Leftarrow l(x_1 - x_0)/d_{01} + x_0 + b(y_1 - y_0)/d_{01}$
23: $y_a \Leftarrow l(y_1 - y_0)/d_{01} + y_0 - b(x_1 - x_0)/d_{01}$
24: $x_b \Leftarrow l(x_1 - x_0)/d_{01} + x_0 - b(y_1 - y_0)/d_{01}$
25: $y_b \Leftarrow l(y_1 - y_0)/d_{01} + y_0 + b(x_1 - x_0)/d_{01}$
26: $d_{2a} \Leftarrow \sqrt{(x_a - x_2)^2 + (y_a - y_2)^2}$
27: $d_{2b} \Leftarrow \sqrt{(x_b - x_2)^2 + (y_b - y_2)^2}$
28: if $d_{2a} > d_{2b}$ then
29:     $x \Leftarrow x_b$
30:     $y \Leftarrow y_b$
31:     $d_2' \Leftarrow d_{2b}$
32: else
33:     $x \Leftarrow x_a$
34:     $y \Leftarrow y_a$
35:     $d_2' \Leftarrow d_{2a}$
36: end if
37: $diff \Leftarrow (d_2 - d_2')/2$
38: $factor \Leftarrow diff/d_2' + 1$
39: $temp \Leftarrow \sqrt{(x_2 - x)^2 + (y_2 - y)^2}$
40: $P_x \Leftarrow x_2 - temp \cdot factor$
41: $P_y \Leftarrow y_2 - temp \cdot factor$
```

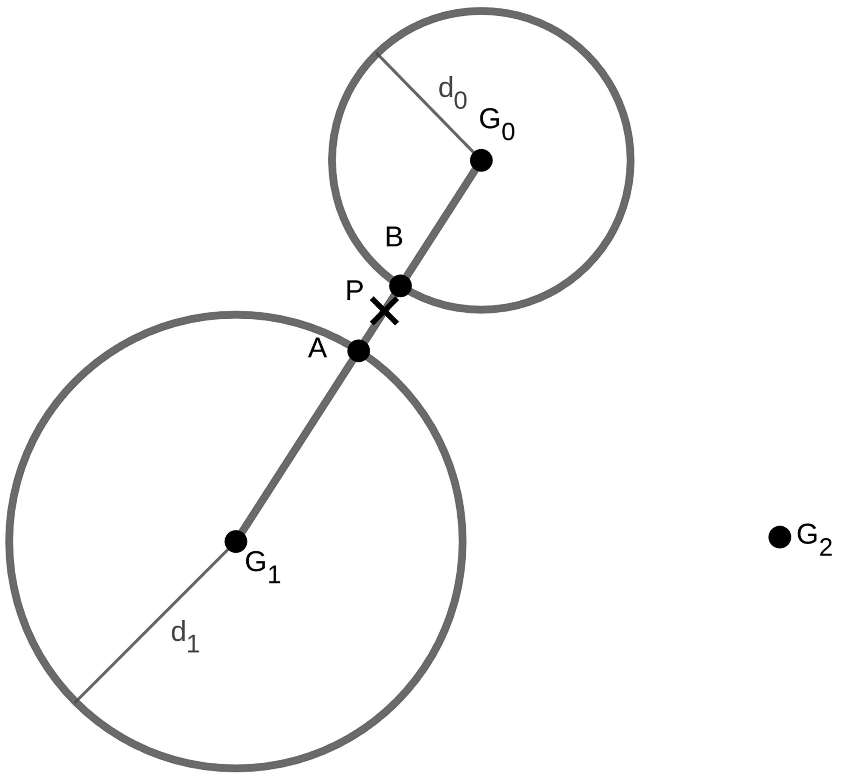


FIGURE 4 | Two gateways received the packet, and the two circles intersect in two different points.

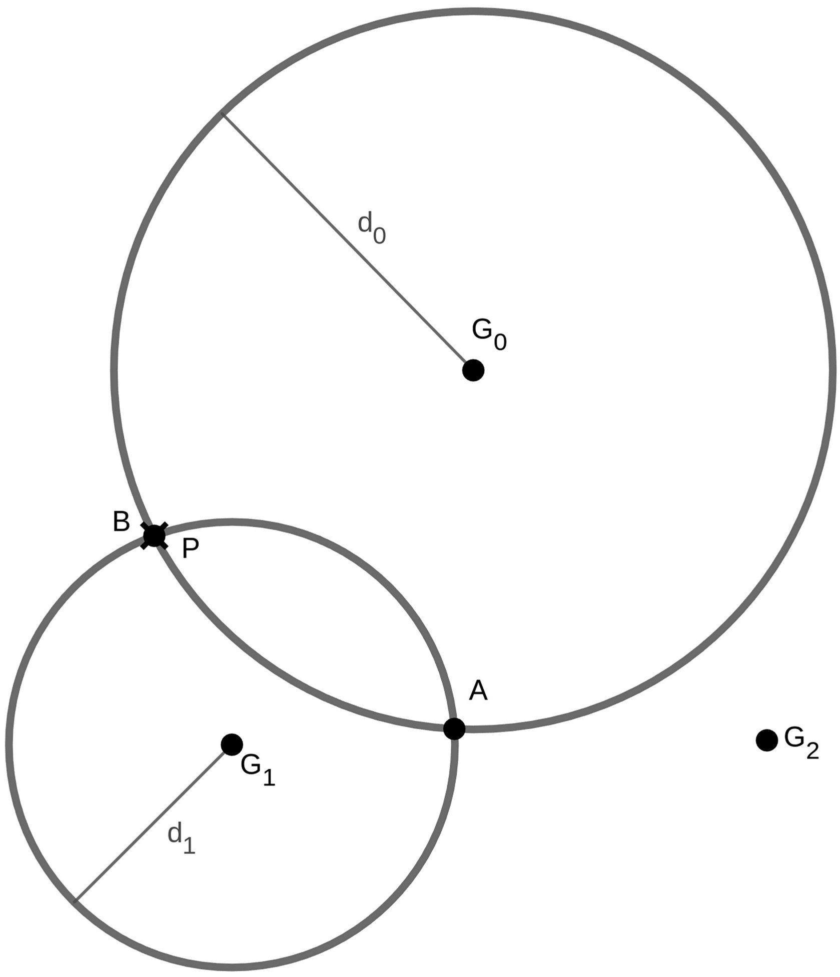


FIGURE 5 | Two gateways received the packet, but the two circles do not intersect.

ALGORITHM 3 | Estimate node position with two gateways.

```
Algorithm 3 Estimate node position with two gateways
Input: $G_0, G_1, G_2$                ▷ Assuming only $G_0$, $G_1$ received the packet
Input: $d_0, d_1$                                          ▷ Calculated distance
Output: $P$                                       ▷ Estimated position of node
1: $C_0 \Leftarrow Circle(G_0, d_0)$          ▷ Circle with centre $G_0$ and radius $d_0$
2: $C_1 \Leftarrow Circle(G_1, d_1)$          ▷ Circle with centre $G_1$ and radius $d_1$
3: $L_{01} \Leftarrow Line(G_0, G_1)$                      ▷ Line comes from $G_0$, $G_1$
4: if $Intersect(C_0, C_1)$ then                            ▷ Circles intersect
5:     $A, B \Leftarrow IntersectionPoints(C_0, C_1)$
6:     if $Distance(A, G_2) > Distance(B, G_2)$ then
7:         $P \Leftarrow A$                            ▷ Farthest point from $G_2$ is $A$
8:     else
9:         $P \Leftarrow B$                            ▷ Farthest point from $G_2$ is $B$
10:    end if
11: else                                             ▷ Circles does not intersect
12:    $A \Leftarrow IntersectionPoint(C_0, L_{01})$
13:    $B \Leftarrow IntersectionPoint(C_1, L_{01})$
14:    $F \Leftarrow d_1/(d_0 + d_1)$
15:    $P_x \Leftarrow A_x + F \cdot (B_x - A_x)$          ▷ Calculate an intermediate point
16:    $P_y \Leftarrow A_y + F \cdot (B_y - A_y)$
17: end if
```

ALGORITHM 4 | Estimate node position with one gateway.

```
Algorithm 4 Estimate node position with one gateway
Input: $G_0, G_1, G_2$                    ▷ Assuming only $G_0$ received the packet
Input: $d_0$                                               ▷ Calculated distance
Output: $P$                                       ▷ Estimated position of node
1: $C_0 \Leftarrow Circle(G_0, d_0)$          ▷ Circle with centre $G_0$ and radius $d_0$
2: $M \Leftarrow (G_1 + G_2)/2$                           ▷ Find median of $G_1$, $G_2$
3: $L_{M0} \Leftarrow Line(M, G_0)$                         ▷ Line comes from $M$, $G_0$
4: $A, B \Leftarrow IntersectionPoints(C_0, L_{M0})$
5: if $Distance(A, M) > Distance(B, M)$ then
6:     $P \Leftarrow A$                               ▷ Farthest point from $M$ is $A$
7: else
8:     $P \Leftarrow B$                               ▷ Farthest point from $M$ is $B$
9: end if
```

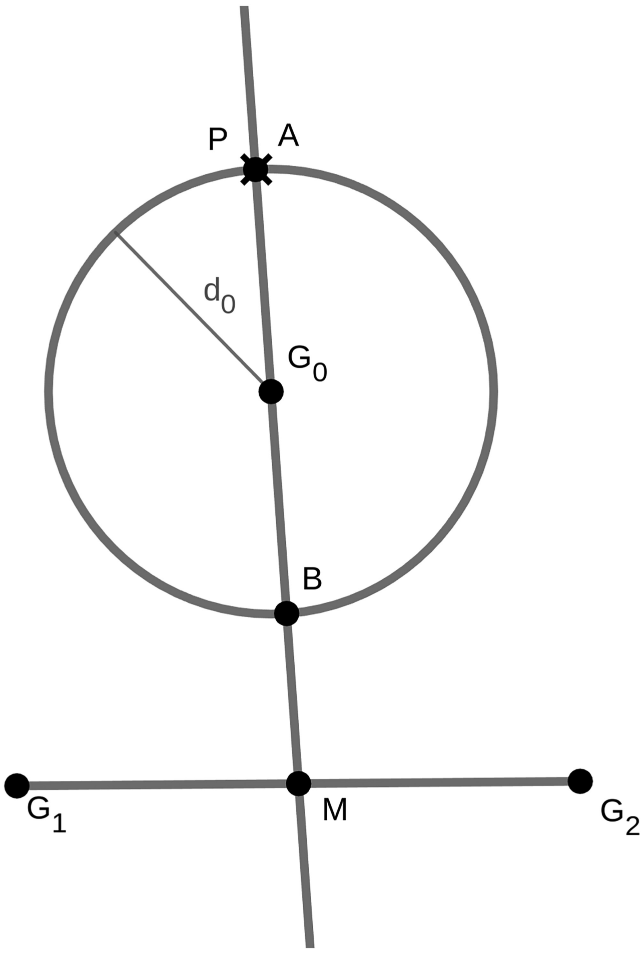


FIGURE 6 | One gateway received the packet.

additional hardware and energy consumption is not substantial, but becomes considerable when dealing with low-cost end devices with restrictions like LoRa devices. For these reasons, some alternative methods have been proposed for positioning in wireless sensor networks.

The two main categories are based on the received signal strength indicator (RSSI) and the time difference of arrival (TDoA). Both methods can be used to estimate the distance of the end device from the gateway, and subsequently, to determine its position in 3D or 2D space.

Regarding LoRa, most research uses RSSI, while some studies also test TDoA [41]. Additionally, most of the research efforts in LoRa are applied to static end devices, with only a few attempting to estimate the position of mobile end devices [42].

More specifically, TDoA relies on the time at which the signal is received by the gateways. Therefore, an additional mechanism is required to synchronize the clocks of all gateways, which can be challenging. Nevertheless, some research uses this method to determine the position of LoRa end devices [43], [41].

Regarding RSSI-based methods, research efforts are divided into two subcategories: those that use the fingerprinting method [44], [45] and those that use trilateration or multilateration to calculate the position of the end device [46], [47]. Of course, in both methods, it is important to address accuracy challenges and apply mitigation techniques, as RSSI measurements are inherently noisy due to signal attenuation, multipath effects, interference, and weather conditions. To tackle this issue, numerous research efforts have proposed various techniques to improve accuracy. The simplest and most common approach is averaging RSSI values from multiple previous messages, which works well for static end devices. Another popular method is using the Kalman filter [48], which is a well known algorithm used to estimate the state of a dynamic system from noisy measurements. Most advanced methods are based on machine learning techniques [49], [50]. The accuracy of the applied methods is quantitatively evaluated using metrics such as mean absolute error (MAE) and root mean square error (RMSE).

After refining the RSSI values, the first step is to estimate the distance between the transmitting end device and the gateways, followed by estimating its position. The distance can be estimated using the log-distance path loss model, which relates RSSI to distance by considering factors such as TP, path loss at a reference distance, and the environmental path loss exponent.

In the fingerprinting method, a reference map of known positions and their corresponding RSSI values is initially constructed. This map serves as a database for positioning, allowing the system to estimate the location of end devices by comparing real-time RSSI measurements with the stored reference data.

On the other hand, when applying trilateration or multilateration to estimate the position of an end device in 2D space, signals must be received from at least three gateways. For 3D positioning, a minimum of four gateways is required to ensure accurate localization.

For example, the most common approach in 2D space is to solve (1), where G_0, G_1, G_2 are the three gateways and d_0, d_1, d_2 are the estimated distances from the end device. For 3D space, an additional equation is required based on the signal from the fourth gateway.

$$\begin{aligned}(X-X_{G_0})^2+(Y-Y_{G_0})^2&=d_0^2\\(X-X_{G_1})^2+(Y-Y_{G_1})^2&=d_1^2\\(X-X_{G_2})^2+(Y-Y_{G_2})^2&=d_2^2\end{aligned} \tag{1}$$

TABLE 3 | Simulation parameters.

Number of nodes	20, 100, 200, 300, 400
Number of gateways	1
Simulation area	6 km × 6 km
Node altitude	0 m
Gateway altitude	150 m
Node speed	(1, 8, 16, 24) mps
Gateway speed	0 mps
Initial SF	12 or algorithm depended
Initial TP	14 or algorithm depended
Number of retries	1
Node mobility	RandomWaypointMobility
Node Wait Time	0–1200 s
Time for First Packet	uniform (0 s, 120 s)
Time for Next Packet	60 s + exponential (60 s)
Path loss model	ShadowLogNormal
$d0$	400
γ	2.08
σ	3.57

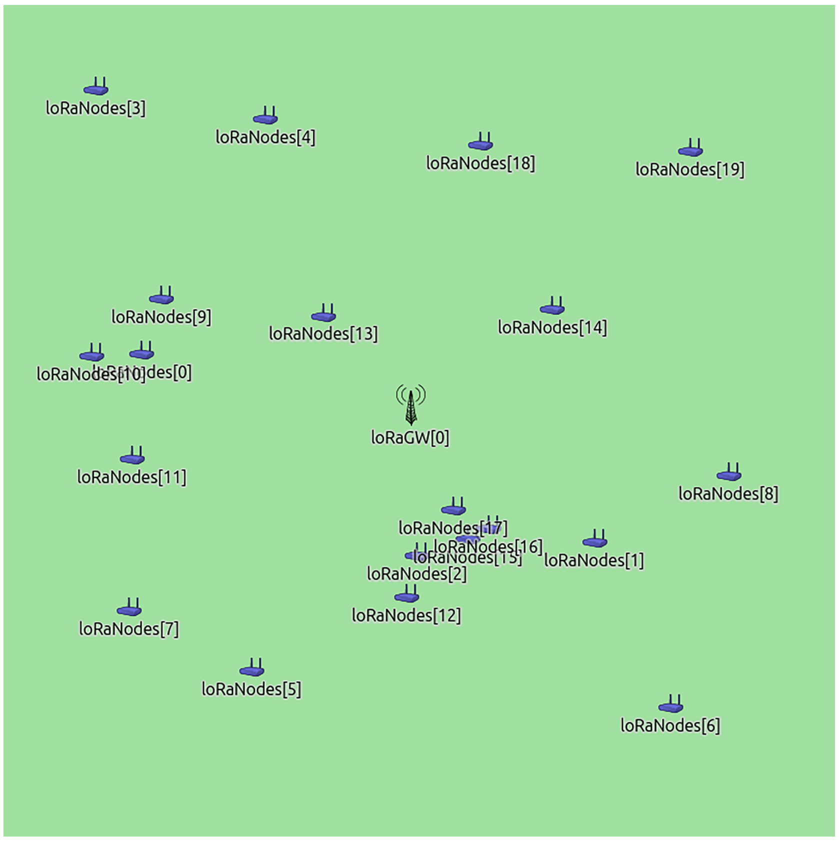


FIGURE 7 | Simulation with a static gateway and mobile nodes around it.

Of course, such an approach introduces significant errors, as transmitted signals often contain substantial interference and noise. As mentioned earlier, factors such as signal attenuation, multipath effects, and environmental variability further impact accuracy. Consequently, solving the problem using (1) is unlikely to yield a unique solution.

3 | Materials and Methods

In this section, we provide detailed information on the methodology we followed. First, we introduce the proposed algorithm for mobile end devices, named M-SADR. Then, we provide information on the software and parameters used to implement our simulations, along with a detailed explanation of the two simulation scenarios used to evaluate the proposed algorithm. Regarding the first simulation, we provide a comprehensive description of the position estimation process for end devices using RSSI, which is used to move the mobile gateways and maintain proximity to the moving group of end devices. In the remainder of the manuscript, the term "end devices" will be used interchangeably with "nodes."

3.1 | Mobile Self Adaptive Data Rate (M-SADR)

To select the optimal values for SF and TP, we propose the M-SADR, which aims to increase the PDR and reduce the energy consumption of mobile end devices. In our method, only the end devices are responsible for adjusting SF and TP, using a lightweight algorithm with low memory requirements and computational complexity.

This approach is particularly important for LoRa end devices, as they are typically constrained in terms of memory and computational capabilities. Furthermore, because most of these devices operate on batteries, it is essential to minimize computational demands and reduce transmission time to extend their lifetime.

In particular, each node maintains an array P of floating-point values, with each element corresponding to one of the available SFs. Each value in the array, ranging from 0 to 1, represents the estimated probability of successful packet delivery when using the associated SF. More precisely, these values are computed as moving averages, continuously updated to reflect the recent success rate of transmissions for each specific SF. Initially, all

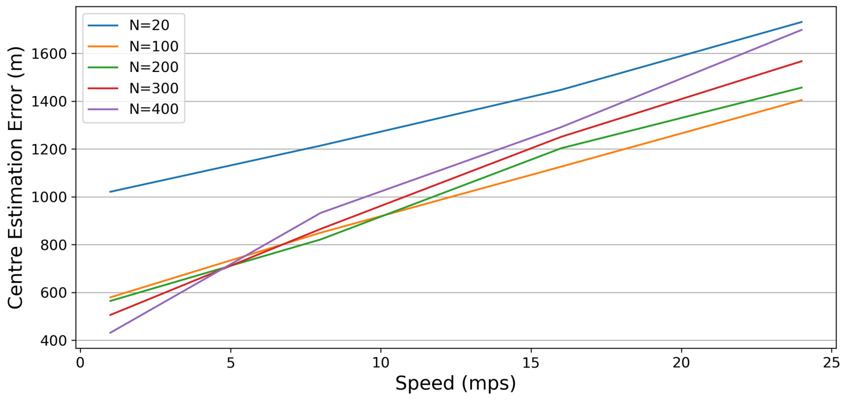


FIGURE 8 | Distance error for various numbers of nodes and speeds. Utilizing M-SADR and RSSI localization.

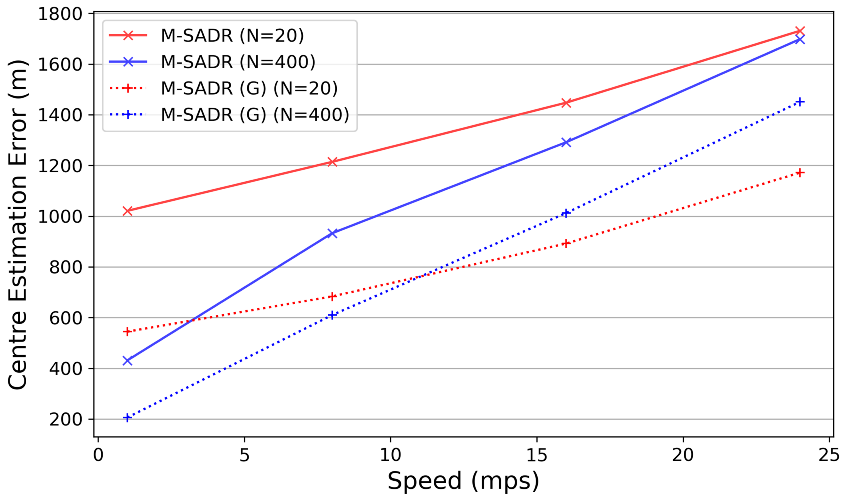


FIGURE 9 | Distance error comparison for M-SADR using RSSI or GNSS localization.

values in $P[SF]$ are set to 1. The sending node always transmits packets with the ACK flag enabled, requiring acknowledgement from the network server. After receiving or not receiving a confirmation, the node updates the table of floating-point values using (2).

$$P[SF] = (1 - b) \times (P[SF]) + b * R_{ACK} \tag{2}$$

where $P[SF]$ is the floating-point value corresponding to the SF, and R_{ACK} is a boolean value equal to 1 when the sending node receives a downlink message and 0 when it does not. The parameter b can be adjusted to control the influence of recent receptions. Lower values of b result in a more stable but slower response, while higher values make it more reactive to changes. Therefore, it is useful to adjust its value based on the success or failure of recent transmissions. In this vein, we define parameter b as shown in equation 3, where its value is set to the minimum of 0.05 or the number of failed transmissions in the last ten packets, multiplied by 0.05.

$$b = \min \begin{cases} 0.05, \\ 0.05 \times \text{Number of } R_{ACK} = \text{false} \end{cases} \tag{3}$$

More specifically, the equation returns a value for b between 0.05 and 0.50. Thus, when there are many successful transmissions in the last ten packets, the value of b is lower, leading to slower changes in $P[SF]$. Conversely, when there are many failed transmissions, the value of b is close to 0.50, resulting in faster changes to $P[SF]$.

When a node needs to send a packet, it selects the SF by finding the maximum value of P. This ensures that the packet is sent with the most appropriate SF based on recent transmissions. In addition, to further reduce energy consumption while aiming to maintain the PDR, the sending node can also adjust the TP. More specifically, it reduces the TP by 2 dBm after two consecutive successfully delivered packets with the same SF, or increases the TP by 2 dBm after two consecutive undelivered packets with the

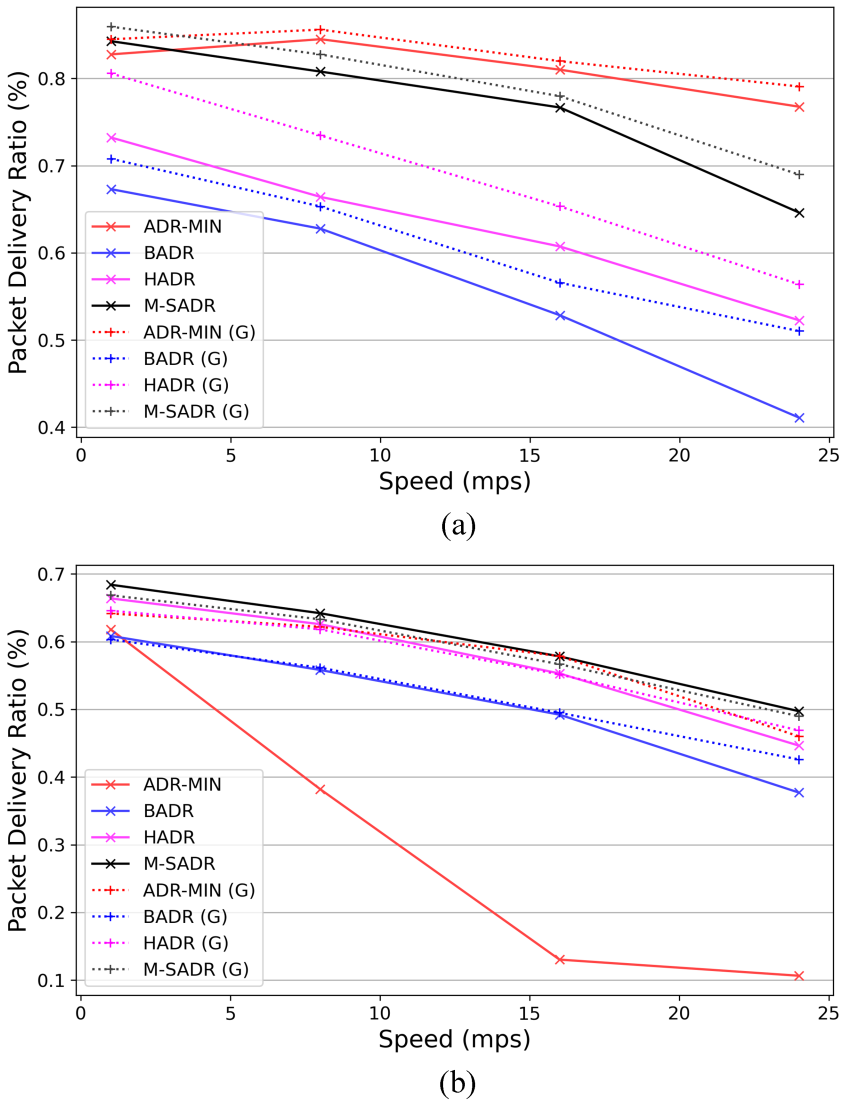


FIGURE 10 | Packet delivery ratio (PDR) for (a) low traffic scenario (20 nodes) and (b) high traffic scenario (400 nodes).

same SF. The minimum TP value used is 10 dBm, and the maximum TP value is 14 dBm.

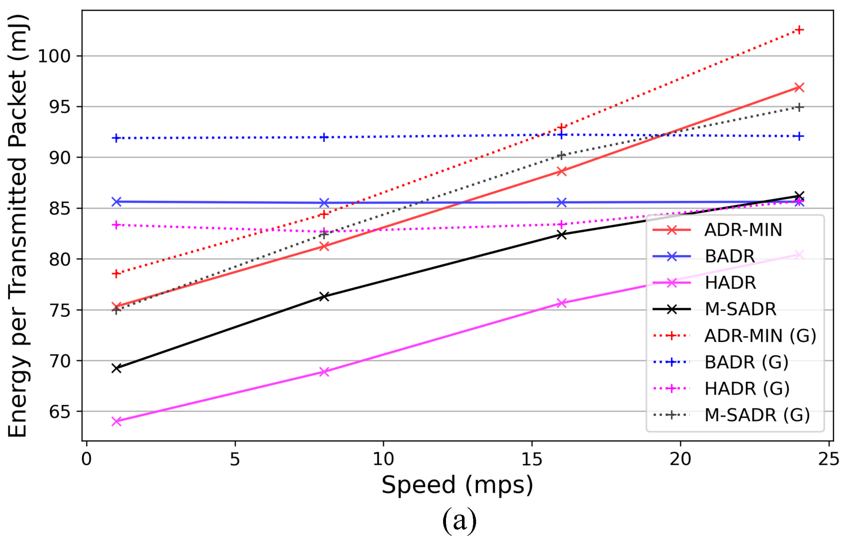
Additionally, when two consecutive transmissions fail at the same SF with a TP of 14 dBm, equation 2 is applied to all $P[SF]$ with an SF different from the current one, assuming that a successful transmission ($R_{ACK} = 1$) would occur with a different SF. This process helps avoid the continued use of the same SF more quickly in cases where no successful transmissions have occurred.

Finally, to achieve a reduction in the SF value for consecutive successful packets, when TP is 10 dBm for two consecutive successful transmissions with the same SF, the node reduces the SF by 1, unless it already has the minimum value, and updates $P[SF - 1]$ with the value of $P[SF]$.

The proposed M-SADR algorithm is shown in Algorithm 1.

As stated earlier, the M-SADR algorithm is designed with computational and memory efficiency in mind, making it highly suitable for deployment on resource-constrained end devices. Its memory usage is minimal, requiring storage for a small set of variables, including a probability array for the SFs, a few integers for state-tracking counters, and auxiliary variables. In particular, the array $P[SF]$ consists of six floating-point numbers, requiring 24 bytes of memory. Other variables, such as the integer counter *cntACK*, the parameter b, and the variable P_{max}, require only a few additional bytes. Thus, the total estimated memory footprint is less than 50 bytes, ensuring compatibility with memory constrained devices.

In terms of computational complexity, the algorithm operates in constant time, $O(1)$, with only a few steps required for each packet transmission. All operations, for selecting the optimal SF and adjusting TP, are performed over fixed-size data structures and do not scale with network size or other system conditions. This ensures constant and low energy consumption, which is critical for energy-constrained devices, particularly those using LoRa wireless technology, which is designed for low energy usage.



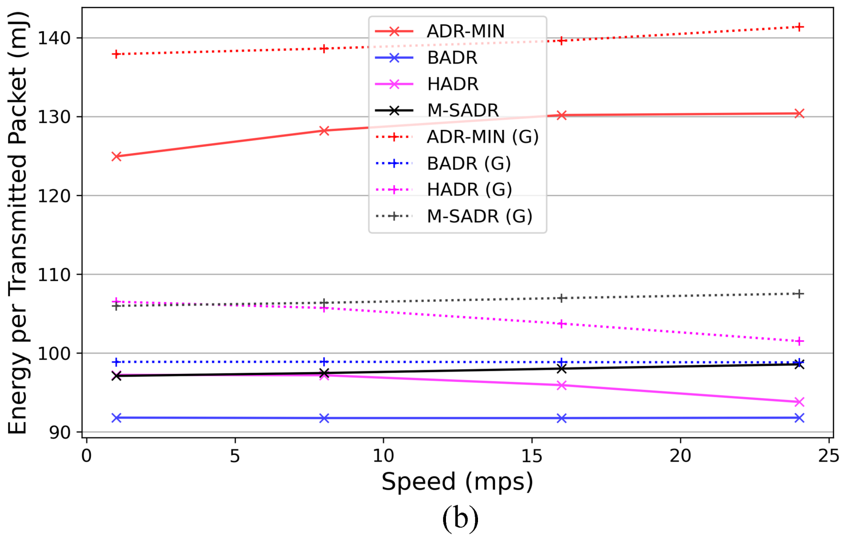


FIGURE 11 | Average consumed energy per transmitted packet (ETP) for (a) low traffic scenario (20 nodes) and (b) high traffic scenario (400 nodes).

3.2 | Simulation Environment

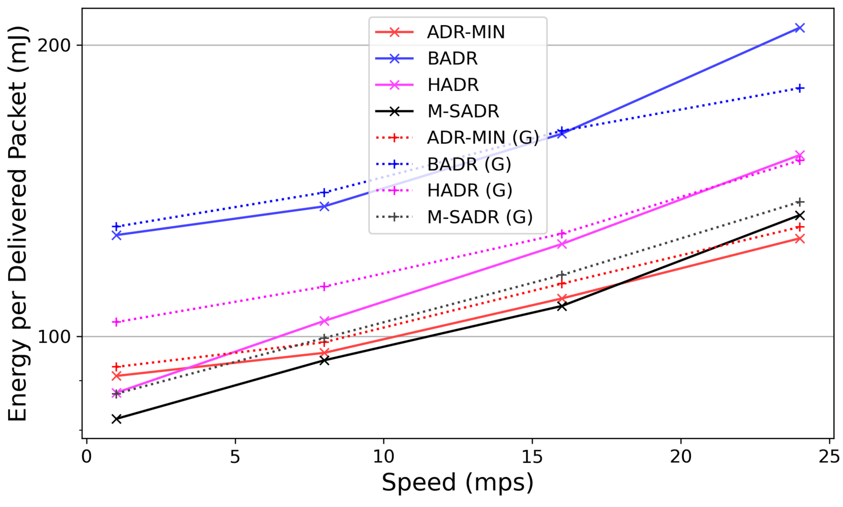
For the simulation environment, we use OMNET++ v6.0.2, a discrete event simulator designed for building and evaluating networks. Additionally, we use INET v4.4.0, a model library for OMNET++ that provides multiple protocols, agents, and other features to construct wired, wireless, or mobile networks. Finally, we use the FLoRa v1.1.0 framework [26], which is suitable for implementing simulations for LoRa networks. More specifically, FLoRa v1.1.0 implements a single-channel LoRaWAN network and offers features such as the configuration of key LoRa parameters, including SF, TP, CR, and bandwidth. In addition, it includes an energy consumption module as well as multiple models for path loss. We have updated the original version to comply with the LoRaWAN v1.1 specifications.

We conducted two different simulations to evaluate the proposed M-SADR algorithm. The first simulation involves a moving group of end devices and three mobile LoRa gateways, mounted on UAVs, following their movement to support connectivity. The second simulation includes a static gateway and multiple mobile end devices moving arbitrarily in the surrounding area.

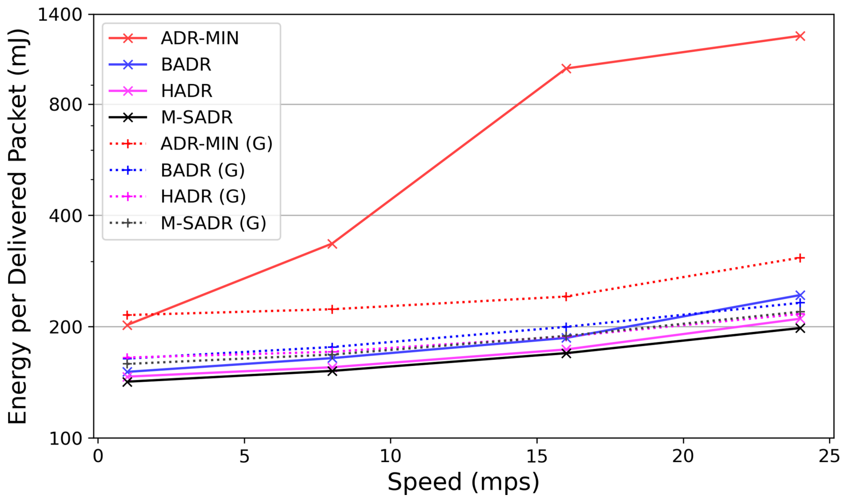
3.3 | Moving Group of Nodes With Mobile Gateways

The first simulation includes a group of nodes moving arbitrarily within an area of 50 km × 50 km. We assume the nodes are moving on a flat area with an altitude of 0 m. The number of nodes in our simulations varies from 20 to 400. To implement the movement, we use the RandomWaypointMobility model for the first node, while the other nodes follow its movement using the SuperpositioningMobility model. This model combines two mobility submodels: AttachedMobility and RandomWaypointMobility. In addition, all nodes are placed in a circle with a radius of 3 km, centered on the first node.

The RandomWaypointMobility model for the first node is configured to move arbitrarily within the 50 km × 50 km area, making random stops with durations between 0 and 1200 s. The



(a)



(b)

FIGURE 12 | Average consumed energy per delivered packet (EDP) for (a) low traffic scenario (20 nodes) and (b) high traffic scenario (400 nodes).

speed is fixed for each simulation, varying between from 1 to 24 mps, as described in the following paragraphs. For the other nodes, the RandomWaypointMobility model is adjusted to move around the group's center with a relative speed of 0.5 mps and random stops lasting between 0 and 600 s.

Regarding the mobile LoRa gateways, we assume they are mounted on UAVs and flying at a height of 150 m above the ground. We use three LoRa gateways, whose movements are managed by the network server. After receiving a packet from a node and calculating the new center of the moving group, the network server sends orders to the gateways to move to their new positions.

Figure 1 illustrates a close-up of the moving group consisting of twenty nodes and the three gateways.

For the path loss model, we use the generic ShadowLogNormal model included in the FLoRa framework, which is suitable for rural areas. Equation (4) describes the ShadowLogNormal model in relation to the distance between the sending node and the gateway.

$$PL_d = PL_0 + 10\gamma \log_{10}\frac{d}{d_0} + X_\sigma \tag{4}$$

where PL_d is the path loss for distance d, PL_0 is the path loss at a reference distance d_0, γ is the path loss exponent and X_σ follows a normal distribution and represents the noise between the transmitter and the receiver.

We have to note that the ShadowLogNormal model, and other similar models, includes factors like free space path loss, environmental effects (such as terrain, buildings, and trees), multipath fading, and obstructions like walls or mountains.

Furthermore, we use SF12 as the initial SF and set the TP to 14 dBm unless specified otherwise by the method. To evaluate network performance under heavy load, only one transmission channel is used, although the LoRaWAN specifications define multiple channels. Additionally, we set the number of retries per packet to one, which is the default value in the LoRaWAN v1.1 specifications. The simulation parameters used are summarized in Table 2.

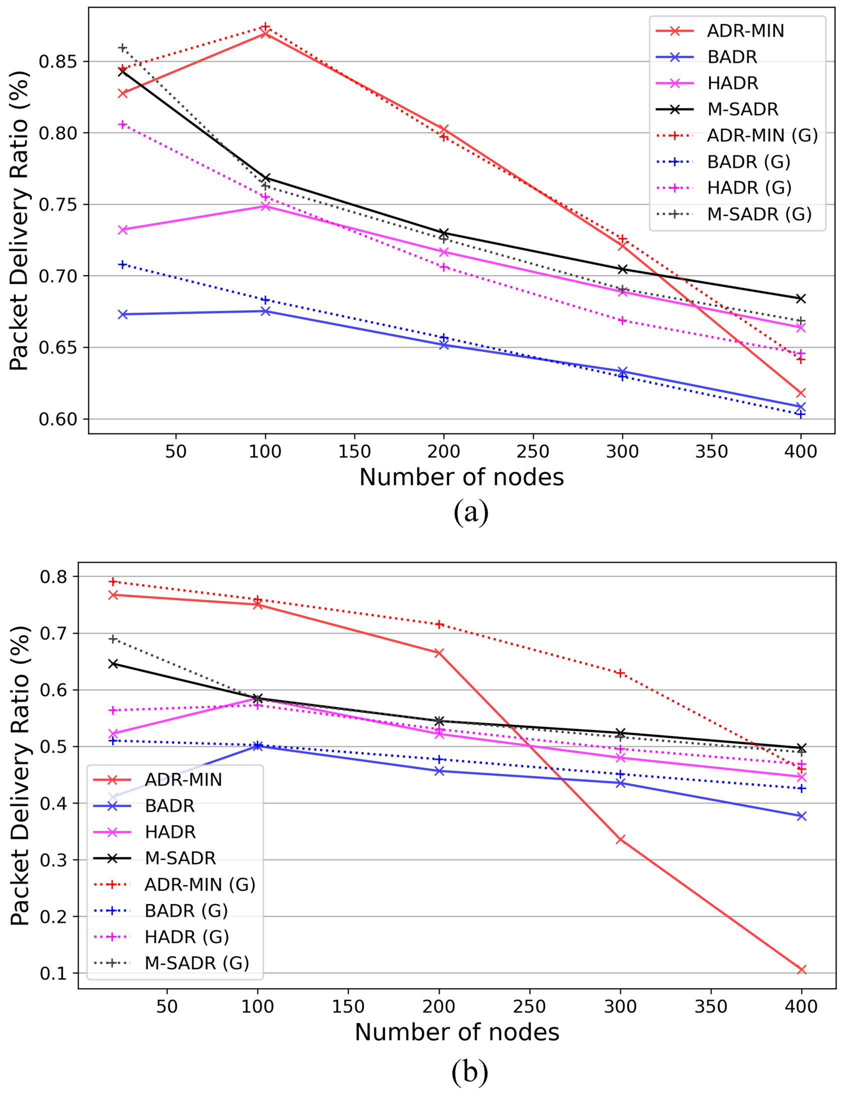


FIGURE 13 | Packet delivery ratio (PDR) for different number of end devices at (a) speed = 1 mps and (b) speed = 24 mps.

3.4 | Movement of Gateways

As described previously, managing the movement of the gateways requires calculating the center of the moving group of nodes. The network server is responsible for this task. More specifically, each time the network server receives a packet from one or more gateways, it first estimates the position of the sending node and then calculates the center (X_c, C_y) of the moving group based on equation 5.

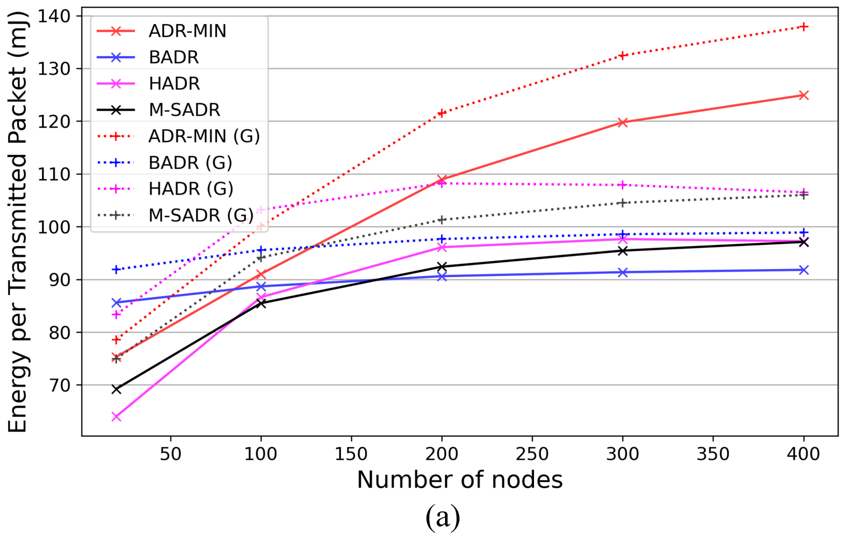
$$X_c = \frac{1}{N} \sum_{i=1}^{N} X_i, \quad Y_c = \frac{1}{N} \sum_{i=1}^{N} Y_i \tag{5}$$

where X_c, Y_c are the calculated coordinates of the center of the moving group, X_i, Y_i are the most recent estimated coordinates of each end device, and N is the number of end devices.

Of course, not every packet is received by all three gateways, so the estimation depends on the number of gateways involved. Additionally, due to interference, RSSI values include significant errors. However, by using information from multiple packets sent by different nodes over short periods, the overall error tends to decrease. Moreover, the methodology does not require a highly precise estimation of the center of the moving group.

In summary, the steps we follow to estimate the position of the sending node are as follows. First, we calculate the distance of the sending node from each gateway that received the packet, using RSSI and (4). We then estimate the position of the sending node based on the number of gateways that received the packet, as described in the following paragraphs.

If the packet is received by all three gateways, we can estimate the position of the sending node using trilateration or similar methods. Due to the errors inherent in RSSI measurements, it is impossible to determine a single precise position; instead, we may obtain a set of possible positions or none at all. For example, Assuming that the calculated distances of the node from the three gateways G_0, G_1, and G_2 are d_0, d_1, and d_2, Figure 2 displays the ideal case where the circles, based on these distances, intersect at a single point. As described above, this is highly unlikely to occur. Figure 3 shows a scenario where all circles intersect, forming a common overlapping area. In this case, the possible position of the sending node could be anywhere within this common area. Of course, other scenarios are also possible.



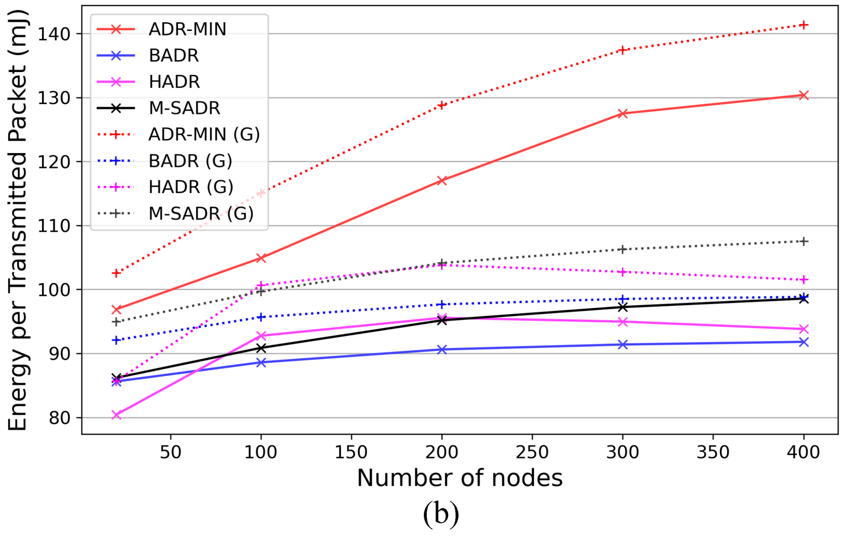


FIGURE 14 | Average consumed energy per transmitted packet (ETP) at (a) speed = 1 mps and (b) speed = 24 mps.

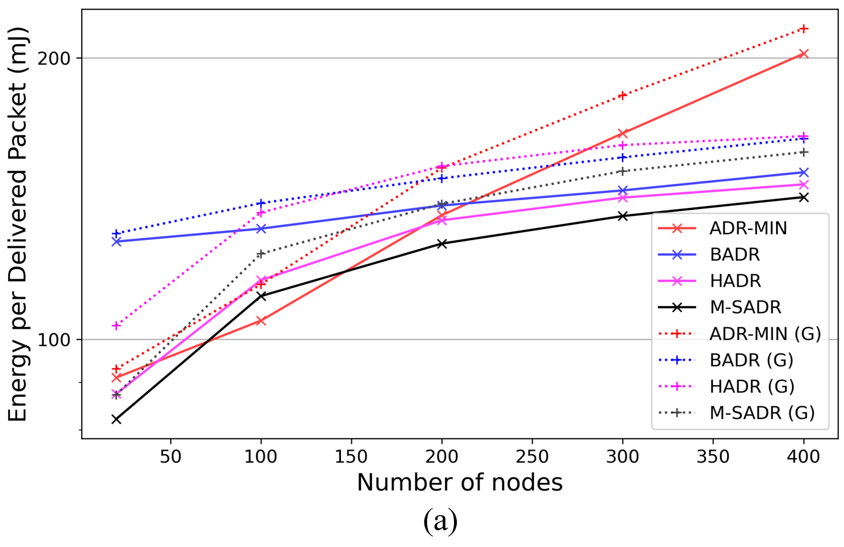
For example, it is possible that all circles intersect but do not form a common overlapping area, or that only two circles intersect, forming a common area. Finally, there may be no intersection among the circles.

To estimate the position of the sending node when the packet is received from three gateways, we have chosen the approach described in [51]. The proposed range error minimization (REM) method is characterized by low computational complexity and offers position estimation with reduced error compared with other methods. Algorithm 2 displays the selected REM method adapted to three gateways for our example.

In case the packet is received by only two gateways, trilateration or similar methods cannot be used to estimate the position of the sending node. Additionally, due to errors in RSSI values, some of the following situations may arise. Assuming that only G_0 and G_1 received the packet of the sending node, and the corresponding calculated distances are d_0 and d_1, the two circles with these radii may intersect at a single point A. In this case, the estimated position P of the node is the intersection point A. However, this ideal scenario is highly unlikely due to errors in RSSI values. Figure 4 displays the situation where the two circles intersect at two points, A and B. In this case, we select the position of the sending node P as the intersection point that is farthest from the third gateway, which, in our example, is point B. Although this is the most pessimistic choice, it helps to keep the gateways closer to the moving group. Of course, it is possible that the circles do not intersect, as illustrated in Figure 5. In this scenario,

TABLE 4 | Results for high speed and high traffic for mobile gateways.

	PDR	ETP	EDP
ADR-MIN	10.7%	130.39	1223.72
BADR	37.7%	91.81	243.48
HADR	44.6%	93.81	210.13
M-SADR	49.7%	98.57	198.24
ADR-MIN (G)	46.0%	141.36	307.31
BADR (G)	42.6%	98.83	232.01
HADR (G)	46.9%	101.52	216.56
M-SADR (G)	49.0%	107.54	219.54



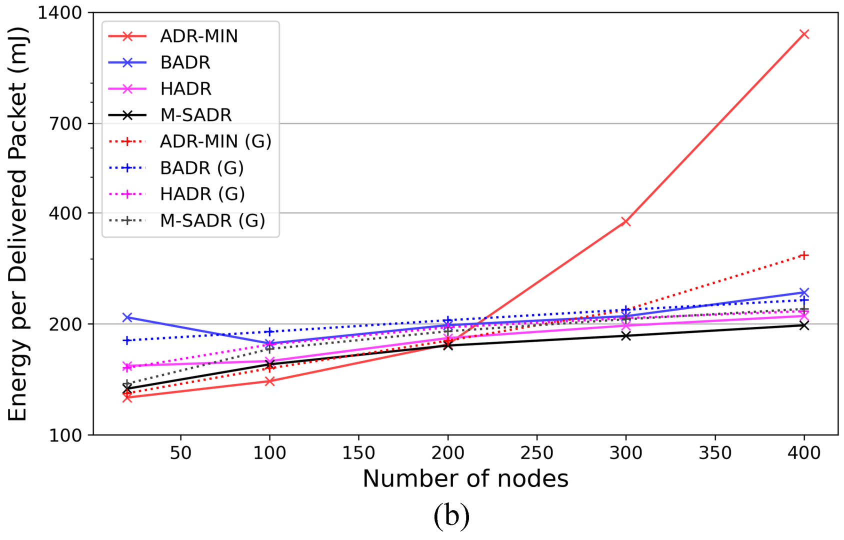


FIGURE 15 | Average consumed energy per delivered packet (EDP) at (a) speed = 1 mps and (b) speed = 24 mps.

we first calculate the intersection points of the circles with the line passing through points A, and B, which is drawn from the centers of the circles, and then select the intermediate point P. It is also possible that one circle is entirely within the other, but this situation is handled in the same way.

The algorithm 3 displays the procedure for estimating the position of the sending node based on RSSI measurements from two gateways. In this example, we assume that only G_0 and G_1 received the packet from the sending node.

The last case is when the packet is received by only one gateway. For example, assuming that the packet is received by gateway G_0 and the calculated distance is d_0, Figure 6 displays the method used to estimate the position of the sending node. First, we calculate the midpoint M between the positions of the other two gateways, G_1 and G_2, and then we use the line defined by M and G_0. The intersection of this line with the circle centered at G_0 with radius d_0 yields two points, A and B. The point farthest from M is chosen as the estimated position P of the sending node. As before, this is also the most pessimistic selection as it is the farthest point from the other two gateways, but it helps to keep the gateways closer to the moving group.

The algorithm 4 displays the procedure to estimate the position of the sending node based on RSSI measurements only from one gateway. In this example, we assume that only G_0 received the packet from the sending node.

After estimating the position of the sending node, we calculate the center of the moving group by finding the average of the positions of the last $N/2$ nodes that successfully sent a packet, where N is the number of nodes of the moving group. We prefer to use only the most recent $N/2$ packets, as information from older packets may lead to an inaccurate position estimate at the current time.

Next, we use the calculated position of the center of the moving group to calculate the new positions of the mobile flying gateways. The specified positions form an equilateral triangle around the center of the moving group, with each gateway positioned at a distance depending on the speed of the moving group of nodes. Thus, the new positions of the gateways are given by (6), (7), and (8).

$$G_0 = \left(C_x, C_y - \frac{2R_{gw}}{3}\right) \tag{6}$$

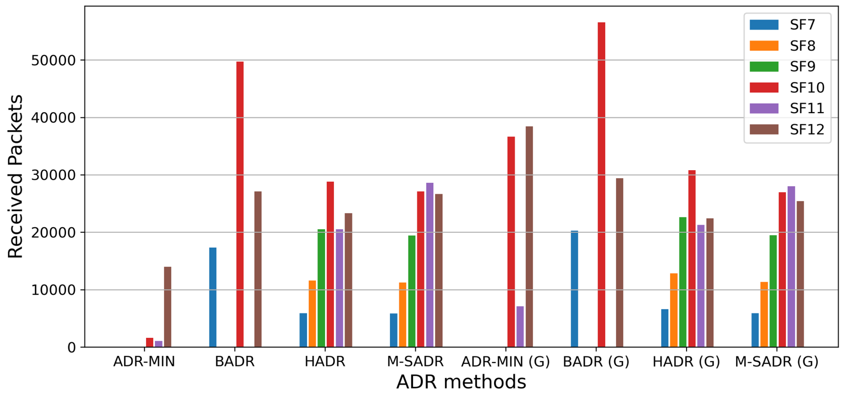


FIGURE 16 | Spreading factor (distribution) across all ADR methods (speed = 24 mps, $N = 400$).

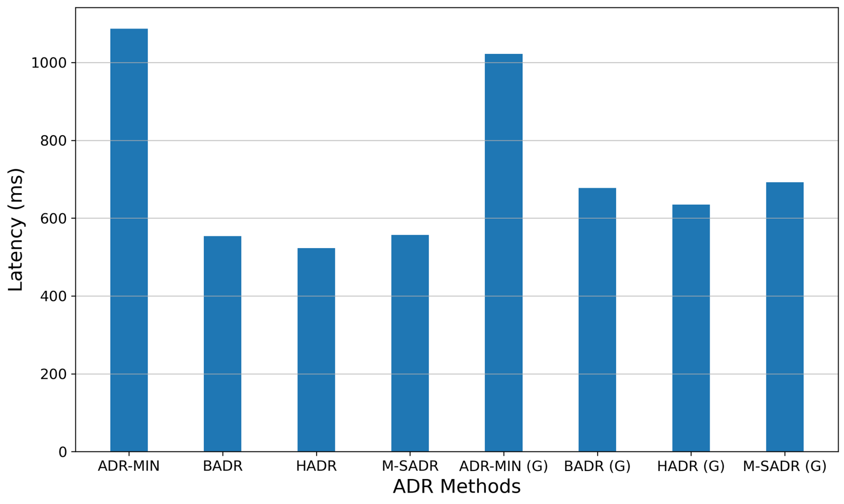


FIGURE 17 | Average latency for all ADR methods (speed = 24 mps, $N = 400$).

TABLE 5 | Packet delivery ratio per spreading factor for mobile gateways.

		SF7	SF8	SF9	SF10	SF11	SF12	Total
ADR-MIN	SP	—	—	—	9796	5260	140,820	155,876
	RP	—	—	—	1586	1051	13,992	16,629
	PDR	—	—	—	16.2%	20.0%	9.9%	10.7%
BADR	SP	124,813	—	—	83,201	—	41,630	249,644
	RP	17,304	—	—	49,733	—	27,096	94,133
	PDR	13.9%	—	—	59.8%	—	65.1%	37.7%
HADR	SP	46,508	46,788	46,590	46,248	28,397	33,404	247,935
	RP	5915	11,596	20,505	28,825	20,508	23,290	110,639
	PDR	12.7%	24.8%	44.0%	62.3%	72.2%	69.7%	44.6%
M-SADR	SP	39,736	40,177	40,311	40,102	39,439	39,266	239,031
	RP	5829	11,220	19,433	27,120	28,613	26,640	118,855
	PDR	14.7%	27.9%	48.2%	67.6%	72.6%	67.8%	49.7%
ADR-MIN (G)	SP	—	—	—	54,680	8494	115,277	178,451
	RP	—	—	—	36,650	7118	38,425	82,193
	PDR	—	—	—	67.0%	83.8%	33.3%	46.1%
BADR (G)	SP	124,712	—	—	83,129	—	41,599	249,440
	RP	20,287	—	—	56,562	—	29,403	106,252
	PDR	16.3%	—	—	68.0%	—	70.7%	42.6%
HADR (G)	SP	46,604	46,929	47,296	46,564	28,511	32,668	248,572
	RP	6585	12,844	22,588	30,800	21,279	22,426	116,522
	PDR	14.1%	27.4%	47.8%	66.1%	74.6%	68.6%	46.9%
M-SADR (G)	SP	39,757	40,187	40,317	40,004	39,389	39,301	238,955
	RP	5917	11,314	19,477	26,932	28,001	25,407	117,048
	PDR	14.9%	28.2%	48.3%	67.3%	71.1%	64.6%	49.0%

Abbreviations: PDR, packet delivery ratio; RP, received rackets; SP, sent packets.

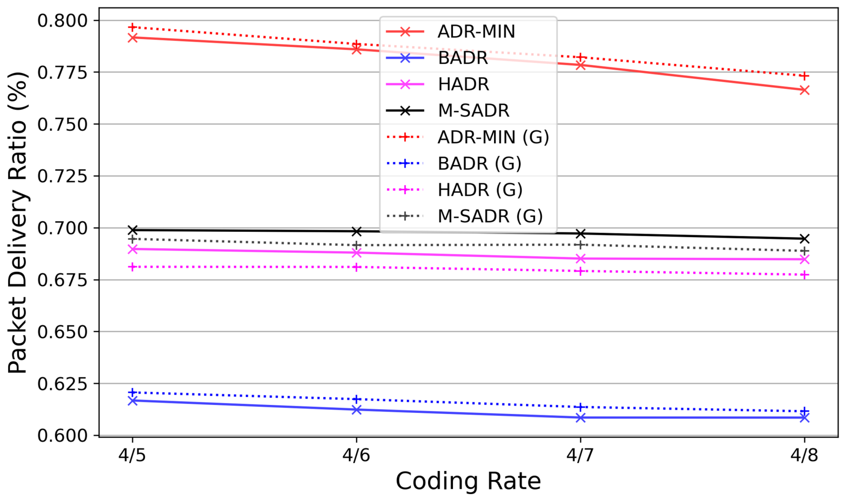


FIGURE 18 | Packet delivery ratio (PDR) for different CR values ($N = 200$, speed = 8 mps).

$$G_1 = \left(C_x + \sqrt{\frac{R_{gw}^2}{3}}, C_y + \frac{R_{gw}}{3}\right) \quad (7)$$

$$G_2 = \left(C_x - \sqrt{\frac{R_{gw}^2}{3}}, C_y + \frac{R_{gw}}{3}\right) \quad (8)$$

where C_x, C_y defines the center of the moving group. R_{gw} is the distance selected from the center of the moving group of nodes, calculated from (9) and depends on the speeds of the moving group.

$$R_{gw} = (1 + (\frac{speed}{maxSpeed})) * R \quad (9)$$

where R is the radius of the moving group, and *maxSpeed* is set to 24 mps in our experiment. Thus, in different simulations, the value of R_{gw} varies and depends on the speed of the moving group. The minimum value of R_{gw} is close to R, which is the radius of the moving group. The maximum value of R_{gw} is equal to $2R$.

3.5 | Mobile Nodes With a Static Gateway

The second simulation we used to evaluate the effectiveness of the proposed M-SADR algorithm is simpler than the first one. It consists of one static gateway and several end devices moving arbitrarily around it. The static gateway is positioned in the center of the simulation area of 6 km × 6 km. The number of nodes in the simulation varies from 20 to 400. In this simulation, the mobile end devices follow the RandomWaypointMobility model, with speeds ranging from 1 to 24 mps. Figure 7 illustrates the setup of the simulation area, with twenty nodes and one gateway.

For the path loss model, we use the generic ShadowLogNormal model included in the FLoRa framework, which is suitable for rural areas. The simulation parameters we use are summarized in Table 3.

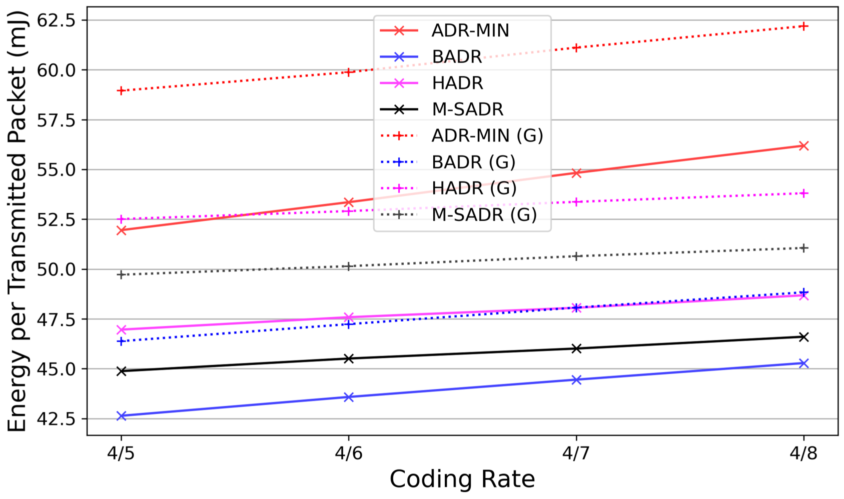


FIGURE 19 | Average consumed energy per transmitted packet (ETP) for different CR values ($N = 200$, speed = 8 mps).

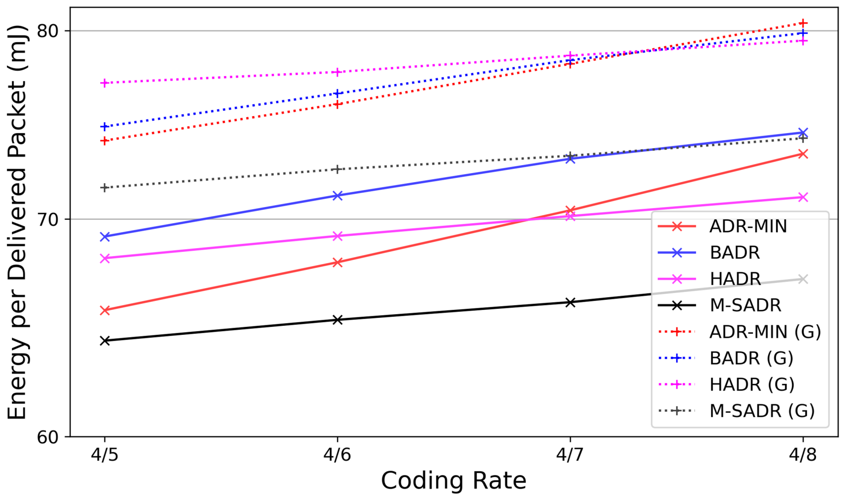


FIGURE 20 | Average consumed energy per delivered packet (EDP) for different CR values ($N = 200$, speed = 8 mps).

4 | Results

In this section, we present the results of the simulation of our proposed method.

In the scenario with mobile gateways and the moving group of nodes, we use all algorithms (ADR-MIN, BADR, HADR, M-SADR) using both RSSI position estimation and GNSS information of the end devices. We must remind the reader that when using GNSS the packet size is larger by 8 bytes as it includes additional information for the position of end devices. On the other hand, as mentioned previously, using RSSI for position estimation may consist of important errors, which may cause difficulties in accurately following the moving group of end devices.

In the scenario with the static gateway and the mobile end devices, we use GNSS information only with the HADR algorithm, while the others do not use it. This is because this scenario does not utilize the position of the end devices, making the inclusion of position information for ADR-MIN, BADR, and M-SADR an unnecessary increase in packet size. However, in the case of HADR, the additional information is used to define the node as static or mobile, which is part of the algorithm to select the appropriate behavior.

In the rest of this section, we refer to the algorithms as ADR-MIN, BADR, HADR, and M-SADR when they do not include GNSS information when sending packets over the LoRa network. Moreover, when including GNSS information, we refer to these methods as ADR-MIN (G), BADR (G), HADR (G), and M-SADR (G), indicating that an additional 8 bytes are included in the packet header.

4.1 | Evaluation for the Moving Group of Nodes With Mobile Gateways

First, we present the results of the method used for controlling the movement of gateways based solely on RSSI signals from the end devices. Second, we present the results of the proposed M-SADR method and compare it with alternative ADR methods,

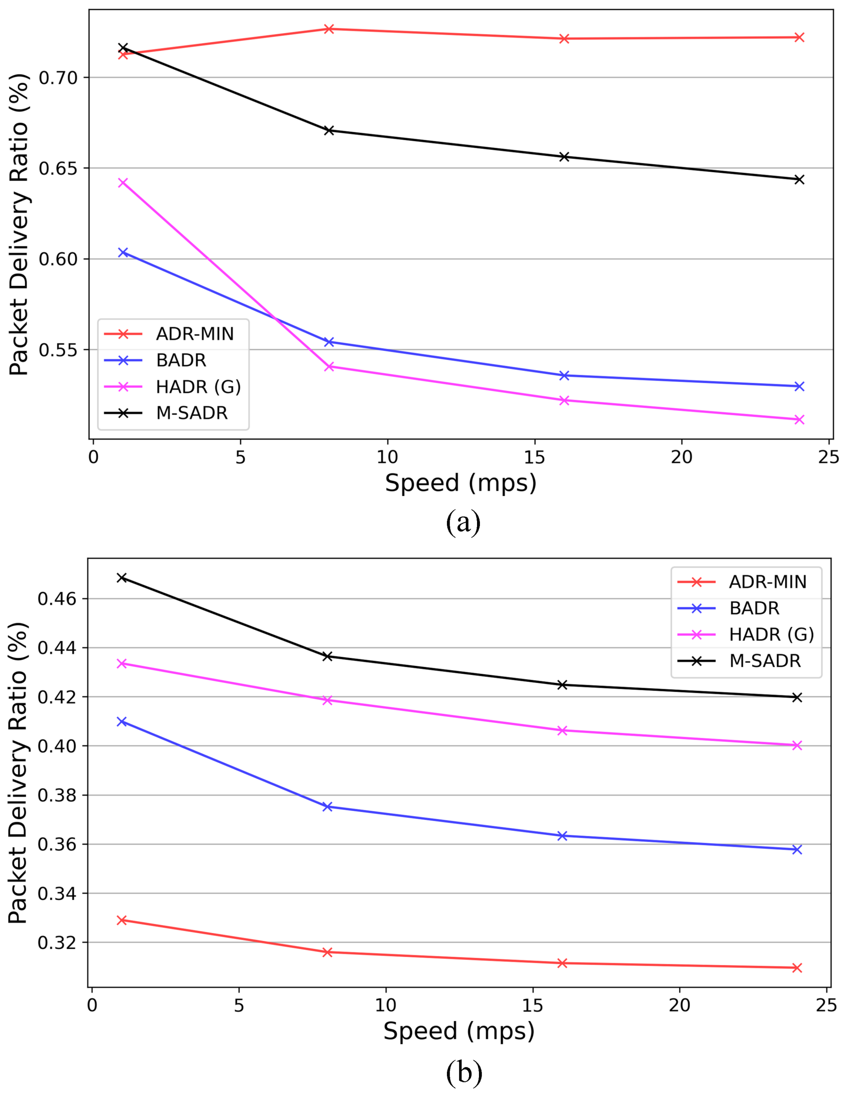
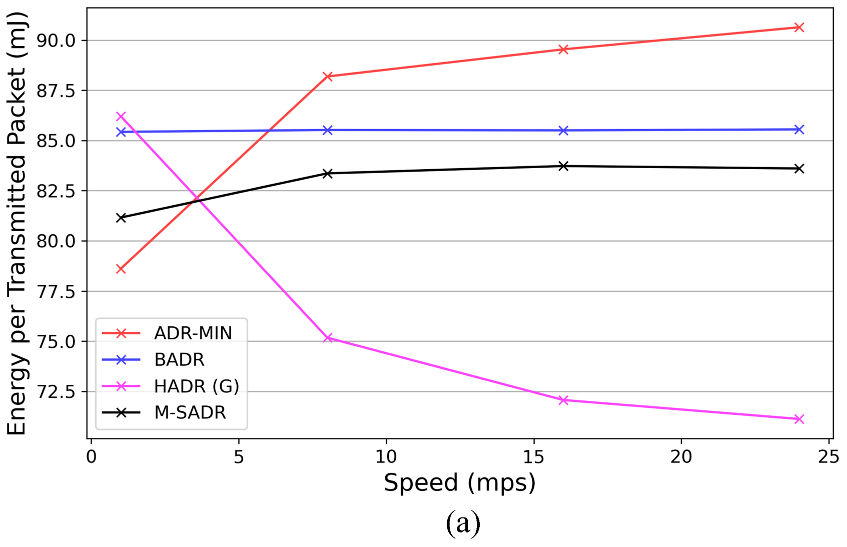


FIGURE 21 | Packet delivery ratio (PDR) with a static gateway for (a) low traffic scenario (20 nodes) and (b) high traffic scenario (400 nodes).

namely ADR-MIN, BADR, and HADR. We also compare the same algorithms (ADR-MIN (G), BADR (G), HADR (G), and M-SADR (G)) when including GNSS information.

Figure 8 displays the error between the calculated center of the moving group of nodes and its actual center. The graph illustrates the error for varying numbers of nodes and speeds, with results based on the simulation using M-SADR. The error tends to increase as the speed of the moving group increases. This is expected because the position estimates are based on previous information, with older positions potentially introducing higher error. This is not only due to errors in position estimation by RSSI, but also the fast movement of end devices. Consequently, the error in the estimated center of the group ranges from about 400–1000 m at low speeds to up to 1.4–1.7 km at high speeds.

To evaluate the effectiveness of the proposed method using the RSSI values, we provide Figure 9, which compares the estimated position of the moving group based on the RSSI information with that using the GNSS information. The graphs are based on simulations with 20 and 400 end devices. Although the GNSS position information is more accurate, it represents an ideal scenario that requires additional hardware for each end device. In addition, it adds an extra 8 bytes to the packet header, resulting in additional power consumption during transmission. In low traffic scenarios (e.g., $N = 20$), GNSS information provides better position estimation, typically around 250 m for low speed (1 mps) and around 500 m at high speed (24 mps). However, the larger packet size for GNSS results in more collisions in higher traffic scenarios (e.g., $N = 400$), leading to slightly worse position estimation for the moving group. This is because it relies only on packets that were successfully delivered but are further back in time. Consequently, the difference between RSSI and GNSS position estimation is around 200 m at low speed (1 mps) and around 300 m at high speed (24 mps). Thus, the RSSI-based estimation, though less precise, is sufficiently accurate for our experiment, as the mobile gateways remain in proximity to the moving group of end devices.



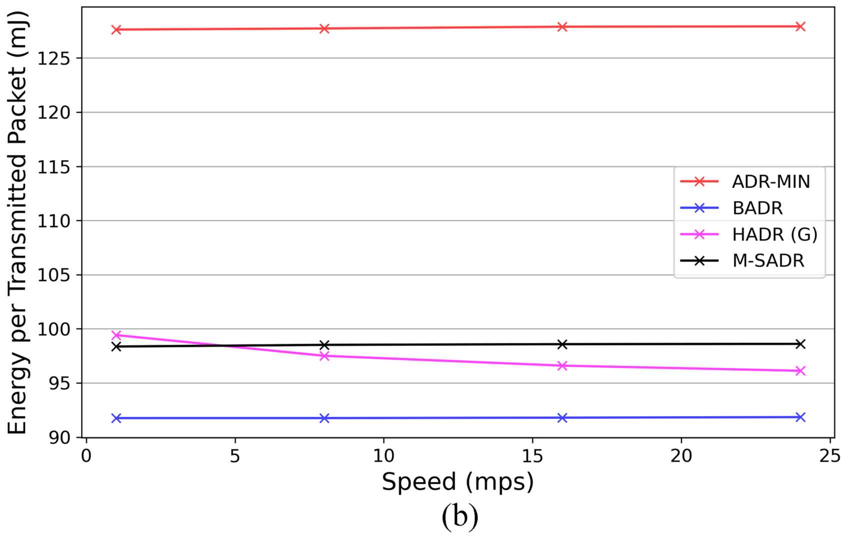
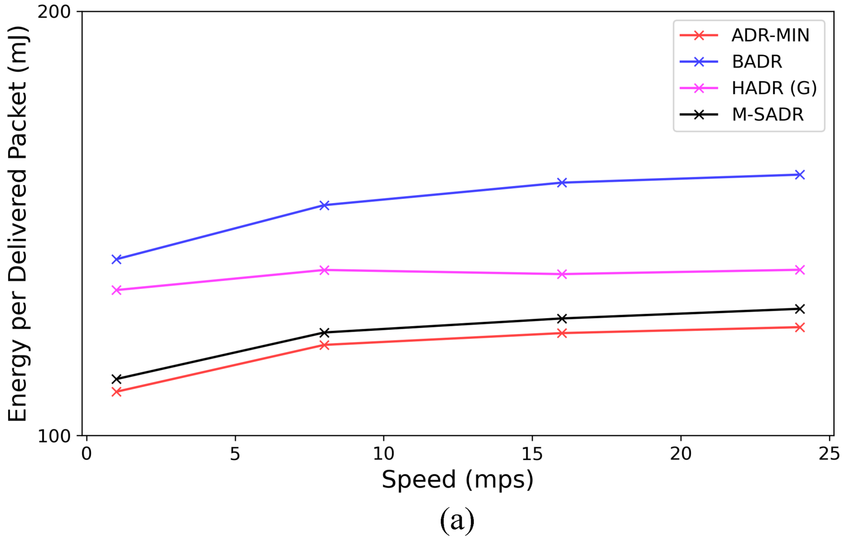


FIGURE 22 | Average consumed energy per transmitted packet (ETP) with a static gateway for (a) low traffic scenario (20 nodes), and (b) high traffic scenario (400 nodes).

Furthermore, to evaluate the effectiveness of the proposed M-SADR method, we compare it with alternative ADR methods, namely ADR-MIN, BADR, and HADR. Figure 10a and Figure 10b compare these methods in terms of PDR. All algorithms are used both with RSSI position estimation (ADR-MIN, HADR, BADR, M-SADR) and with GNSS position information (ADR-MIN (G), BADR (G), HADR (G), M-SADR (G)). More specifically, Figure 10a displays a comparison of the algorithms at different speeds of the moving group with 20 nodes, while Figure 10b presents a comparison with 400 nodes. From Figure 10b, it seems that the M-SADR and M-SADR (G) methods achieve better PDR than the competing methods using RSSI in high-traffic scenarios, and similar to ADR-MIN (G) and HADR (G), which utilize GNSS information. It also performs well in low-traffic conditions, as illustrated in Figure 10a, because only ADR-MIN and ADR-MIN (G) give higher results.

When dealing with LPWANs, evaluating the energy consumption of end devices is crucial, especially the consumed energy per delivered packet (EDP). For that reason, in Figure 11a and Figure 11b, we provide a comparison of the average consumed energy per transmitted packet (ETP) for 20 and 400 nodes, respectively. Moreover, Figure 12a and Figure 12b present the consumed EDP for 20 and 400 end devices, respectively. In essence, Figure 12a is a combination of Figure 10a and Figure 11a, while Figure 12b is a combination of Figure 10b and Figure 11b. The comparison indicates that M-SADR achieves lower energy consumption per successfully delivered packet compared with its competitors across all tested speeds. We have to note that the *y*-axes in Figure 12a and Figure 12b are in logarithmic scale.

To illustrate this more clearly, we present Figure 13, Figure 14, and Figure 15, which display a comparison of the algorithms at speeds of 1 and 24 mps for different numbers of end devices. More specifically, Figure 13 shows a comparison of the PDR, indicating that only ADR-MIN and ADR-MIN (G) perform better than M-SADR in some cases, but they fall short in energy consumption. Figure 14presents the average energy consumption



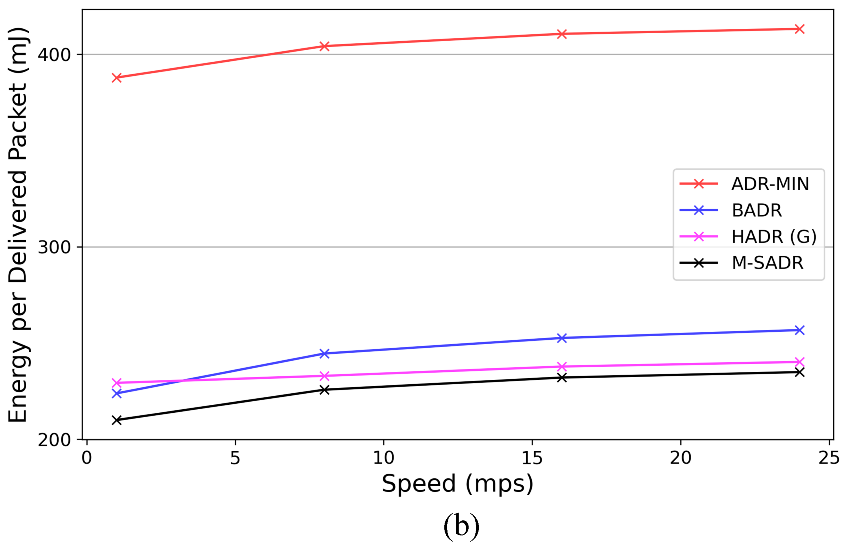


FIGURE 23 | Average consumed energy per delivered packet (ETP) with a static gateway for (a) low traffic scenario (20 nodes) and (b) high traffic scenario (400 nodes).

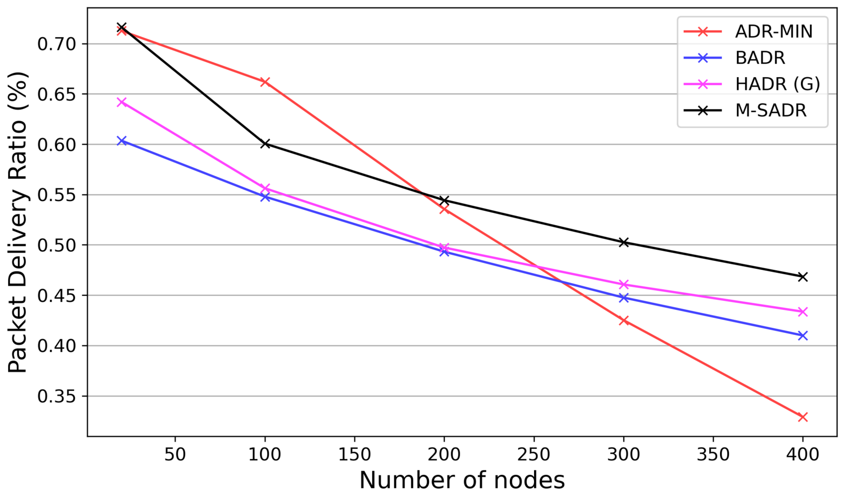
per transmitted packet, where M-SADR shows lower energy usage compared with ADR-MIN and HADR, and is comparable to BADR. However, when considering the energy consumption per delivered packet (Figure 15), M-SADR outperforms all competitors by achieving the lowest values in both low and high traffic scenarios. We have to note that the *y*-axes in Figure 12a and Figure 12b are in logarithmic scale.

Table 4 displays the results for PDR, average ETP, and the average EDP for the simulation with high speed (24 mps) and high traffic ($N = 400$). The results show that M-SADR achieves a PDR of 49.7%, with M-SADR (G) in second place and HADR (G) in third place with a PDR of 46.9%. Moreover, M-SADR demonstrates better energy consumption per successfully delivered packet (198.24 mJ), while HADR (G) ranks third at 216.56 mJ. Thus, M-SADR outperforms its best competitor by 2.8% in PDR and 9.2% in energy consumption (EDP).

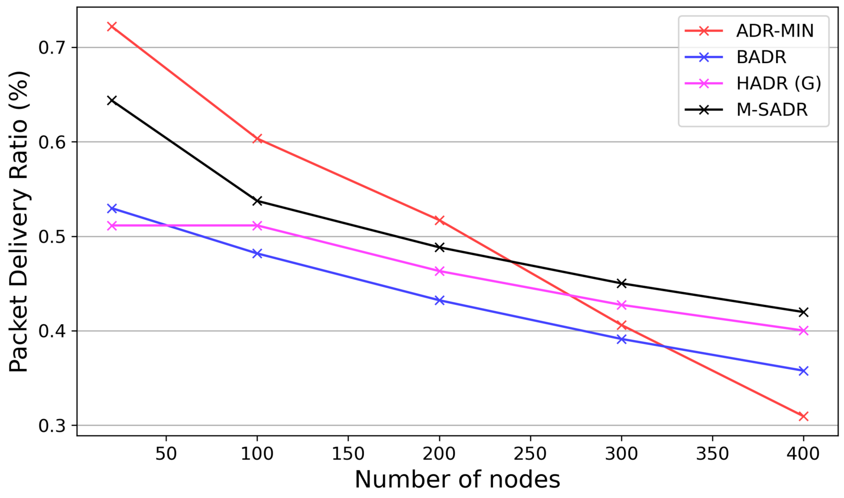
The effectiveness of the proposed M-SADR method is attributed to its optimal distribution of SFs and its ability to adapt rapidly based on network traffic and the positions of end devices. Figure 16illustrates the distribution of SFs for all ADR methods at a speed of 24 mps with 400 end devices.

As observed, M-SADR achieves better distribution by using SFs that are more likely to result in successful transmissions. This approach also helps avoid collisions at the gateway, as it can effectively demultiplex packets with different SFs. In contrast, ADR-MIN tends to use a limited number of SFs for most transmitted packets, leading to a high percentage of collisions at the gateway, especially under high network traffic. BADR, which uses only three SFs (7, 10, and 12), results in increased collisions during high traffic and can also cause a significant number of lost packets when using SF7, or increased energy consumption with SF12, even when not necessary. Lastly, HADR gives comparable results, as it switches through all SFs sequentially, but falls behind in successfully using larger SFs, which results in higher energy consumption and more collisions at the gateways in high traffic networks.

Furthermore, Figure 17 presents the average latency of the delivered packets for all compared ADR methods at a speed of 24 mps



(a)



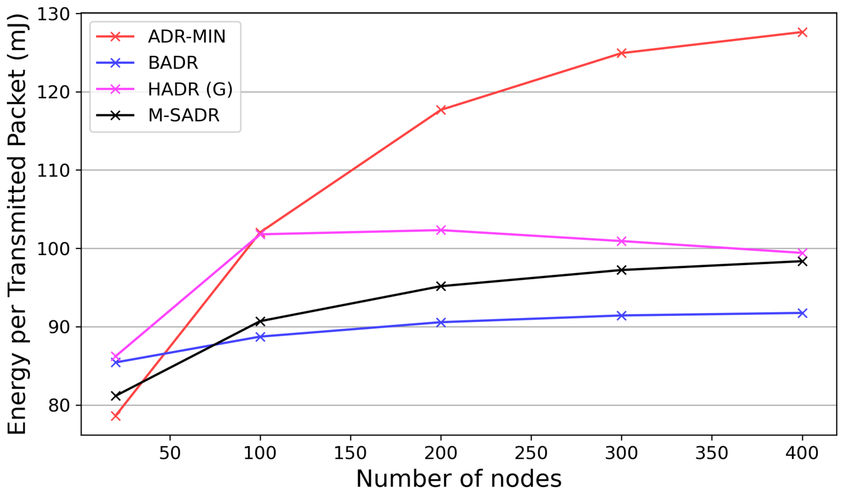
(b)

FIGURE 24 | Packet delivery ratio (PDR) for different number of end devices at (a) speed = 1 mps and (b) speed = 24 mps.

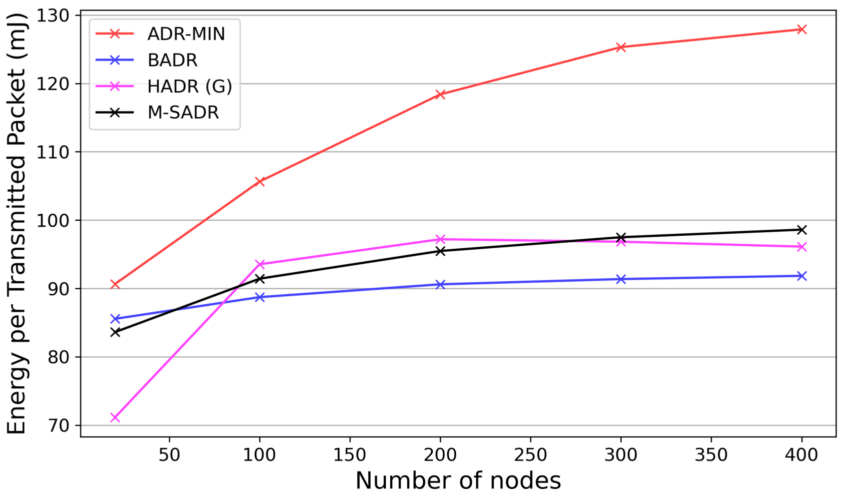
with 400 end devices. The latency calculation is equal to ToA as only one transmission per packet is considered. It appears that ADR-MIN has the highest packet delivery latency. Regarding M-SADR, it demonstrates a comparable average latency to BADR, while only HADR achieves better average packet latency. However, both BADR and HADR perform worse in other metrics discussed above. Additionally, HADR is not the officially proposed algorithm, as the official method is HADR (G), which incorporates GNSS information. Moreover, all methods tested with GNSS information in this scenario exhibit higher average packet latency, which was expected due to the additional 8 bytes in the packet header.

Additionally, Table 5 presents the results per SF for all evaluated algorithms in high traffic scenario. More specifically, for each SF, displays the number of sent packets (SP), the number of received packets (RP), and the PDR. Finally, it displays the summary results for all SFs per algorithm. From that table, we can see the details where M-SADR prevails over the other algorithms. For example, ADR-MIN and ADR-MIN (G) use larger SFs, results in lower PDR due to collisions over the air. In addition, this also results in sending significantly fewer packets due to duty cycle (1%) limitations. BADR and BADR (G) utilize the transmission more effectively as they send a lot of packets with SF7, but this also works as a drawback, as it leads to a low PDR with SF7. Regarding HADR and HADR (G), the distribution of SFs is almost equal, and the difference is attributed to a feature of the algorithm to transmit with the same SF after a lost transmission. Finally, M-SADR and M-SADR (G) achieve better distribution of the SFs, they select the optimal one when needed based on the last transmissions. Moreover, compared with HADR, M-SADR achieve better PDR in almost all SFs.

Finally, to evaluate the effectiveness of the proposed method from a different perspective, we compare it with the alternative methods across all CR values. It is worth noting that CR can take one of the following values: 4/5, 4/6, 4/7, or 7/8, which correspond to the use of 1, 2, 3, or 4 additional bits per 4 useful bits, respectively. The comparison is based on simulations involving 200 end devices moving at a speed of 8 mps. The results for the PDR are presented in Figure 18, where it can be observed that only ADR-MIN achieves a better ratio than M-SADR. Moreover, the PDR slightly decreases for all methods as the CR increases, which is expected, because higher CR values lead to larger packet sizes and, consequently, an increased likelihood of collisions or content corruption.



(a)



(b)

FIGURE 25 | Average consumed energy per transmitted packet (ETP) at (a) speed = 1 mps and (b) speed = 24 mps.

Moreover, Figure 19 presents the average energy consumption per transmitted packet, where M-SADR demonstrates lower energy usage compared with ADR-MIN and HADR, and is comparable to BADR. In contrast, Figure 20 illustrates the average energy consumption per delivered packet, where M-SADR outperforms all competing methods by achieving the lowest values. Furthermore, as the CR increases, M-SADR exhibits a more pronounced relative advantage in energy efficiency compared with the other methods. It is also important to note that the *y*-axes in Figure 19 and Figure 20 are presented on a logarithmic scale.

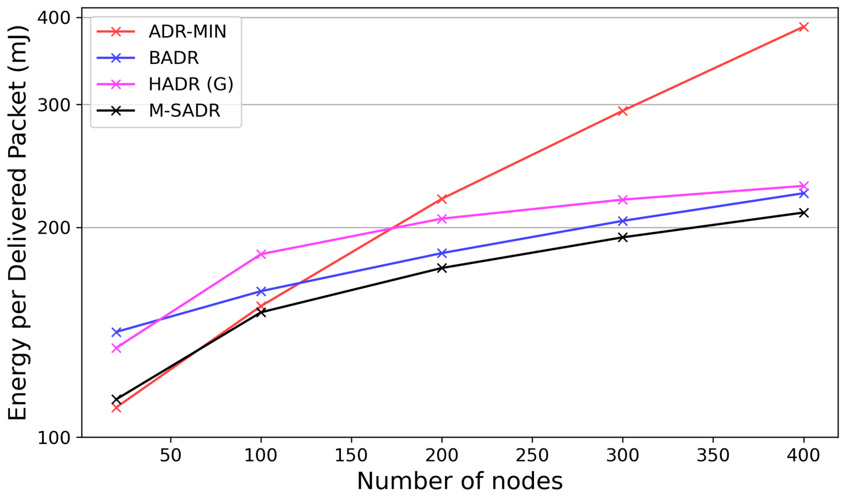
4.2 | Evaluation for the Mobile Nodes With a Static Gateway

The results of the evaluation of M-SADR with a single static gateway are presented in the following paragraphs.

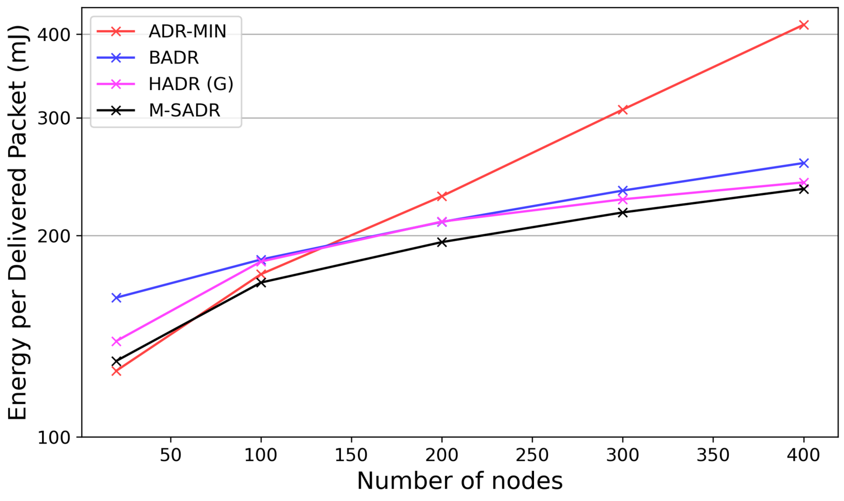
More specifically, to evaluate the effectiveness of the proposed M-SADR method, we compare it with alternative ADR methods, namely ADR-MIN, BADR, and HADR (G). Only HADR (G) includes position information based on GNSS, as it is critical for the algorithm to define the end device as static or mobile. Moreover, position estimation using RSSI is not employed in this simulation, as it is not feasible with only a static gateway. It is also important to note that position estimation was used in the previous simulation to reposition the mobile gateways.

Figure 21a and Figure 21b provide a comparison of these methods regarding the PDR. More specifically, Figure 21a displays a comparison of the algorithms at different speeds of the moving group with 20 nodes, while Figure 10b presents a comparison with 400 nodes. From Figure 21b, it seems that the M-SADR method achieves better PDR than the competing methods in high-traffic scenarios. It also performs well in low-traffic conditions, as illustrated in Figure 21a. In essence, only ADR-MIN achieves better results in low traffic scenarios, but it consumes more energy.

Figure 22a and Figure 22b compare the average consumed ETP for 20 and 400 nodes, respectively. Moreover, Figure 23a and Figure 23b present the average consumed EDP for 20 and 400



(a)



(b)

FIGURE 26 | Average consumed energy per delivered packet (EDP) at (a) speed = 1 mps and (b) speed = 24 mps.

end devices, respectively. In essence, Figure 23a is a combination of Figure 21a and Figure 22a, while Figure 23b is a combination of Figure 21b and Figure 22b. The comparison shows that M-SADR achieves a lower energy consumption per successfully delivered packet compared with its competitors across all tested speeds.

To illustrate this more clearly, we present Figure 24, Figure 25, and Figure 26, which display a comparison of the algorithms at speeds of 1 and 24 mps for different numbers of end devices. More specifically, Figure 24 compares the PDR, showing that M-SADR outperforms its competitors as the number of nodes increases. Figure 25 presents the average energy consumption per transmitted packet, where M-SADR demonstrates lower energy usage compared with ADR-MIN and HADR (G), and is comparable to BADR. However, when considering the energy consumption per packet successfully transmitted packet (Figure 26), M-SADR outperforms all competitors, by achieving the lowest values in both low and high traffic scenarios.

Table 6 displays the results for PDR, average ETP, and the average EDP for the simulation with high speed (24 mps) and high traffic ($N = 400$). The results show that M-SADR achieves a PDR of 42.0%, while HADR (G) ranks second with a PDR of 40.0%. Moreover, M-SADR demonstrates better energy consumption per successfully delivered packet (234.89 mJ), with HADR (G) in second place at 240.16 mJ. Thus, M-SADR outperforms its best competitor by 2.0% in PDR and 2.2% in energy consumption (EDP).

Figure 27 illustrates the distribution of SFs for all ADR methods at a speed of 24 mps with 400 end devices. It appears that M-SADR and HADR (G) achieve better distribution of SFs compared with the other algorithms.

TABLE 6 | Results for high speed and high traffic for a static gateway.

	PDR	ETP	EDP
ADR-MIN	31.0%	127.92	413.16
BADR	35.8%	91.84	256.72
HADR (G)	40.0%	96.12	240.16
M-SADR	42.0%	98.61	234.89

Moreover, Table 7 also displays the distribution of SFs for all algorithms. From the table, we can notice that the M-SADR achieves better PDR for most SFs compared with HADR (G). Moreover, considering that HADR (G) uses an additional 8 bytes for GNSS information to determine whether the device is mobile or static, we can recognize that M-SADR is self-adaptive to any situation of the end device without requiring additional information in the packet, resulting in lower energy consumption.

Finally, Figure 28 presents the average latency of the delivered packets for all compared ADR methods at a speed of 24 mps with 400 end devices. Because only one transmission per packet is used, the latency calculation is equal to ToA. ADR-MIN exhibits the highest packet delivery latency, while M-SADR achieves performance comparable to BADR and HADR.

Finally, to further evaluate the effectiveness of the proposed method, we compare it with the other alternative approaches across all CR values. The comparison is conducted using simulations that involve a single static gateway and 200 mobile end devices moving at a speed of 8 mps. The PDR results are shown in Figure 29, where it can be observed that only ADR-MIN achieves a higher ratio than M-SADR. Additionally, the PDR exhibits a slight decline for all methods as the CR increases, which is expected because higher CR values result in larger packet sizes and, consequently, a greater likelihood of collisions or content corruption.

Moreover, Figure 30 presents the average energy consumption per transmitted packet, where M-SADR demonstrates lower energy usage compared with ADR-MIN and HADR, and is close to BADR. In contrast, Figure 31 illustrates the average energy consumption per delivered packet, where M-SADR outperforms all competing methods by achieving the lowest values. Notably, as the CR increases, the relative improvement of M-SADR becomes more pronounced compared with its competitors. It is also important to note that the y-axes in Figure 30 and Figure 31 are presented on a logarithmic scale.

Overall, M-SADR demonstrates better performance compared with its competitors in terms of power consumption per successfully delivered packet. More specifically, based on Table 4 and Table 6, it achieves 2.4% better in PDR and 5.7% lower power consumption per successfully delivered packet, respectively.

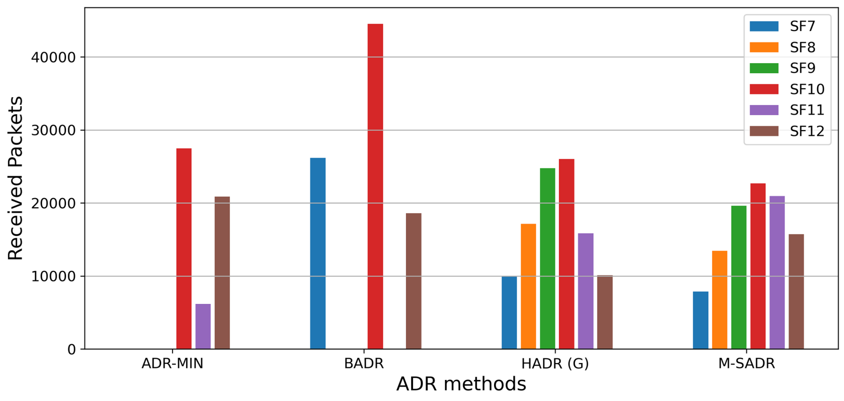


FIGURE 27 | Spreading factor (distribution) across all ADR methods (speed = 24 mps, $N = 400$).

TABLE 7 | Packet delivery ratio per spreading factor for a static gateway.

		SF7	SF8	SF9	SF10	SF11	SF12	Total
ADR-MIN	SP	—	—	—	49,542	8673	117,849	176,064
	RP	—	—	—	27,483	6187	20,851	54,521
	PDR	—	—	—	55.5%	71.3%	17.7%	31.0%
BADR	SP	124,802	—	—	83,190	—	41,621	249,613
	RP	26158	—	—	44,534	—	18,610	89,302
	PDR	21.0%	—	—	53.5%	—	44.7%	35.8%
HADR (G)	SP	51,521	52,225	53,566	51,968	29,662	20,498	259,440
	RP	9996	17,117	24,769	26,028	15,838	10,086	103,834
	PDR	19.4%	32.8%	46.2%	50.1%	53.4%	49.2%	40.0%
M-SADR	SP	39,822	40,097	40,177	39,890	39,406	39,431	238,823
	RP	7847	13,463	19,600	22,688	20,930	15,727	100,255
	PDR	19.7%	33.6%	48.8%	56.9%	53.1%	39.9%	42.0%

Abbreviations: PDR, packet delivery ratio; RP, received packets; SP, sent packets.

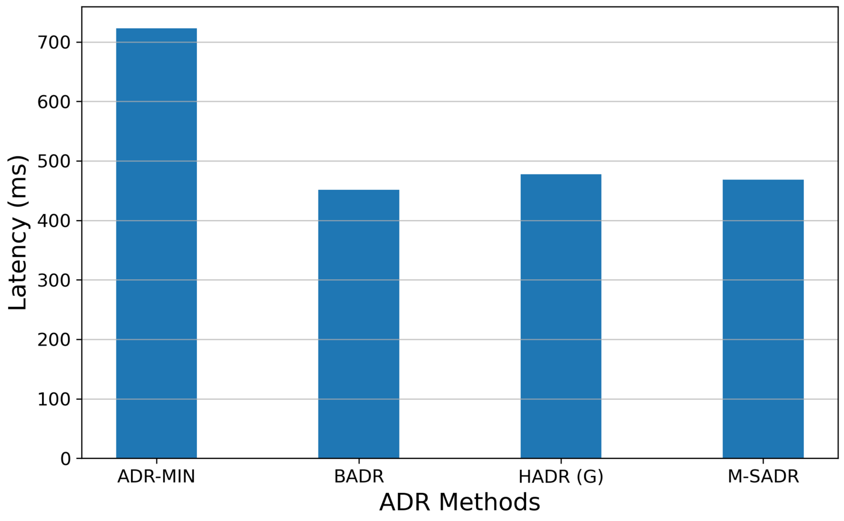


FIGURE 28 | Average latency for all ADR methods (speed = 24 mps, $N = 400$).

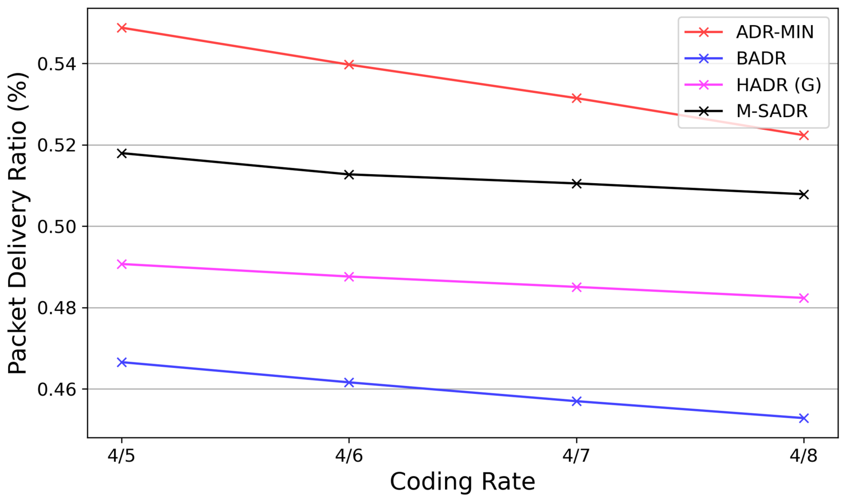


FIGURE 29 | Packet delivery ratio (PDR) for different CR values ($N = 200$, speed = 8 mps).

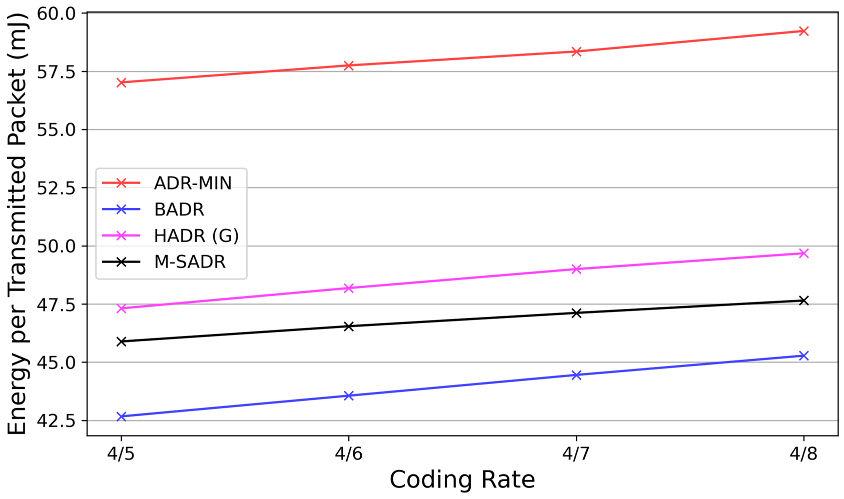


FIGURE 30 | Average consumed energy per transmitted packet (ETP) for different CR values ($N = 200$, speed = 8 mps).

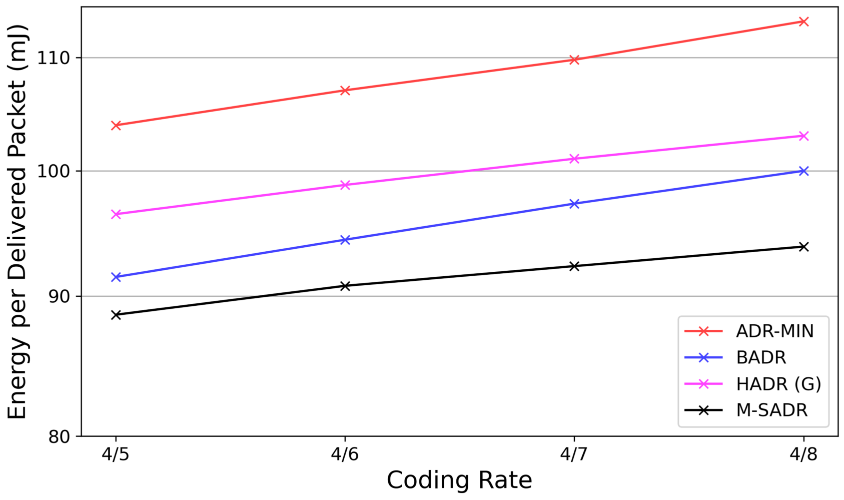


FIGURE 31 | Average consumed energy per delivered packet (EDP) for different CR values ($N = 200$, speed = 8 mps).

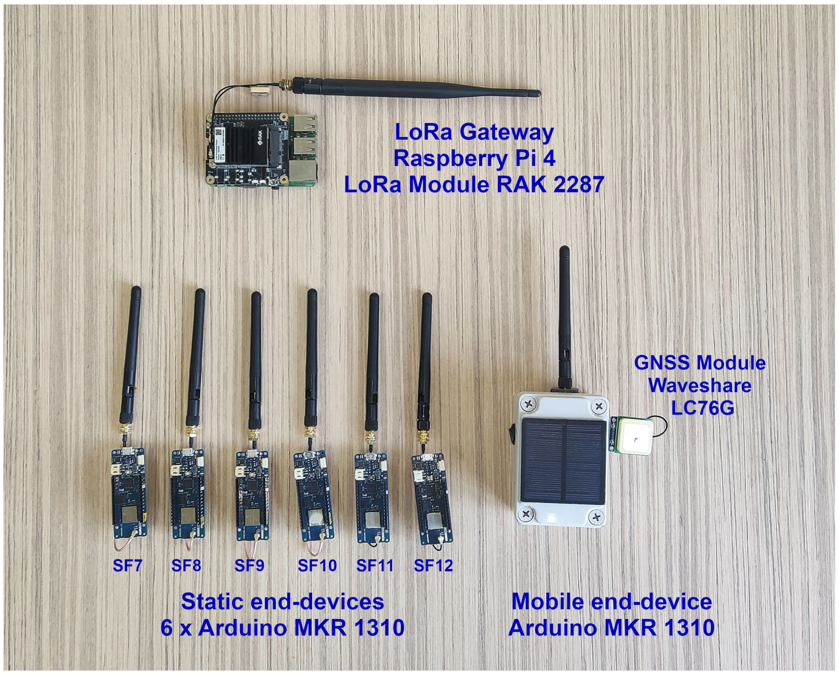


FIGURE 32 | The hardware used for the small scale testbed. A LoRa gateway and seven Arduino MKR 1310.

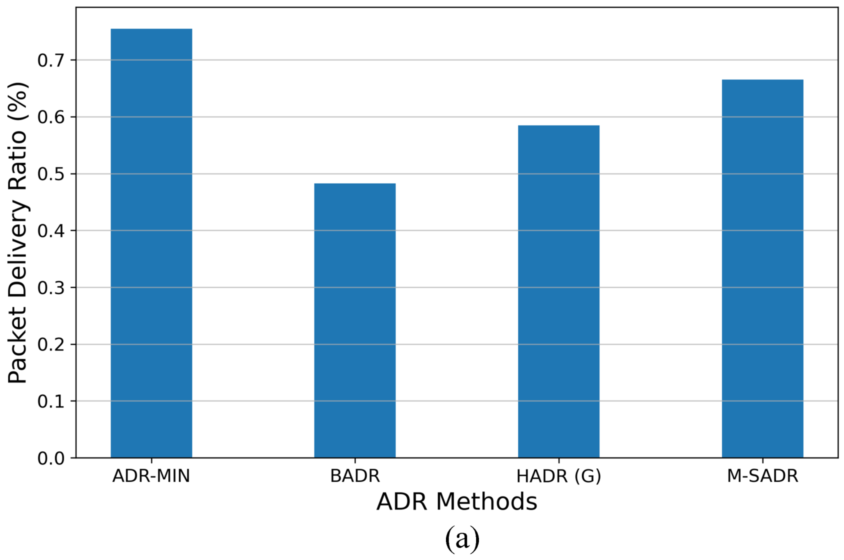
5 | Small Scale Testbed

To evaluate the effectiveness of the proposed M-SADR method, we deployed a small-scale testbed with two different scenarios. The first scenario involves a single mobile end device moving arbitrarily within an area with one gateway and no other end devices. The second scenario includes additional stationary end devices that continuously transmit messages to simulate network congestion.

For the end device, we used the Arduino MKR1310 with a 2 dBi gain 868 MHz antenna, powered by an external Li-Ion 18650 3.7 V battery. It features the powerful Murata CMWX1ZZABZ module, integrated with the SX1276 chipset from Semtech for LoRa modulation. To unlock the full capabilities of the Murata module, we updated the official firmware with the Open LoRaWAN Modem [52]. The firmware was updated using the Arduino IDE and the script provided on the GitHub page. The updated firmware provides an AT command interface that is backward-compatible with Murata's proprietary LoRaWAN firmware. Additionally, it complies with the LoRaWAN v1.1 specifications and removes many of the limitations present in the official firmware.

In our testbed implementation, we disabled the 1% duty cycle limit and used a single channel (868.5 MHz) for transmissions. Although this configuration does not comply with LoRaWAN v1.1 specifications, it was essential for testing the compared ADR methods under heavy load with only a few end devices. We acknowledge that this could impact power consumption and network capacity in a real-world deployment, but this setup was critical for controlled testing of ADR performance. Additionally, the initial SF was set to 10 or depended on the method, and the initial TP was set to 14 dBm for all compared methods. Finally, the number of retransmissions per packet was set to one, in line with the default value in the LoRaWAN v1.1 specifications.

For the LoRa gateway, we used a Raspberry Pi 4 (4 GB model) with the RAK 2287 LoRa module and an antenna with 3 dBi gain. The RAK 2287 is based on the SX1302 chip from Semtech and is connected via a Pi HAT to the Raspberry Pi 4. It supports up to 10 programmable parallel demodulation paths and can



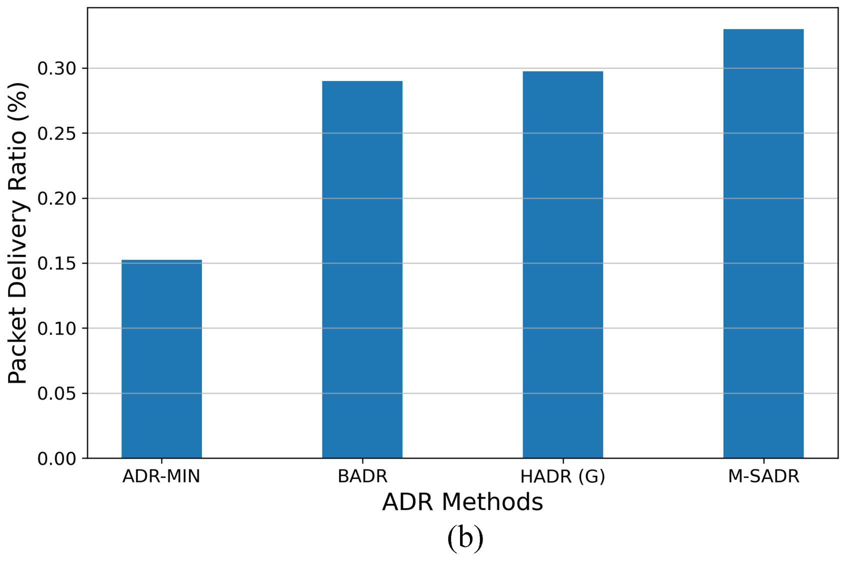


FIGURE 33 | Packet delivery ratio (PDR) in Testbed: (a) low traffic and (b) high traffic.

demodulate up to 16 packets simultaneously. The Raspberry Pi 4 was set up using the official RAK Wireless image, which comes with Raspberry Pi OS pre-installed, along with the necessary software for operating a LoRa gateway. We updated the operating system to the latest version (Raspberry Pi OS v12) and upgraded all included software to their latest stable releases. We configured the LoRa packet forwarder to operate in the EU868 regional band. For the network server, we selected ChirpStack v3.16.8, pre-installed on the RAK Wireless image. It was configured via the configuration file to receive packets on a single channel at 868.5 MHz, which was the designated transmission frequency for the end devices. Additionally, the default ADR method was disabled, and to implement the ADR-MIN and HADR (G) methods on the network server side, we used an MQTT broker to receive packets from the end devices and respond when necessary, adjusting the SF and TP as required. An MQTT broker can be easily implemented in Python using the Paho MQTT library.

The experimental area was located in a park within an urban environment, with dimensions of 250 $m \times 200m$. The area contained several obstacles, such as trees and other objects. The LoRa gateway was placed on a building near the experimental area. It is important to note that no other LoRa networks or competing radio frequency signals were present in the area, minimizing external interference during the tests. We used seven Arduino MKR1310 devices as the end devices. One device was used as the mobile testing end device, which ran the compared ADR methods (MINADR, BADR, HADR (G), M-SADR), and it was configured to send packets every 8 to 12 s. The remaining six end devices were used exclusively in the second scenario and placed as stationary devices within the experimental area. These stationary devices sent packets periodically at random intervals between 4 and 10 s, each assigned a specific SF ranging from SF7 to SF12. The hardware used in the testbed is shown in Figure 32.

The data for each ADR method was collected across two distinct experimental scenarios. In both scenarios, for each ADR method tested, 400 packets were transmitted from the mobile end device. The mobile end device followed a predefined path within the experimental area, moving slowly between designated points. At each point, the device made a brief stop for approximately one minute to simulate real-world intermittent communication and

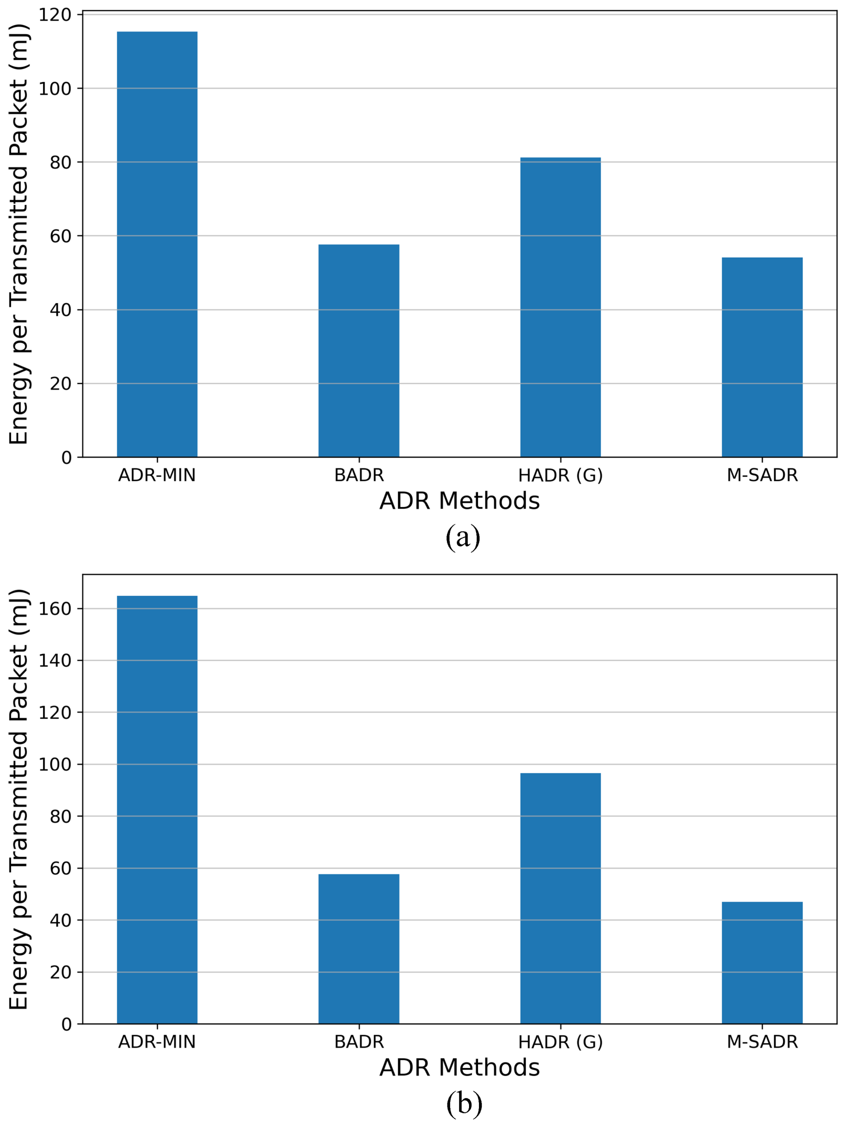


FIGURE 34 | Average consumed energy per transmitted packet (ETP) in testbed: (a) low traffic and (b) high traffic scenario.

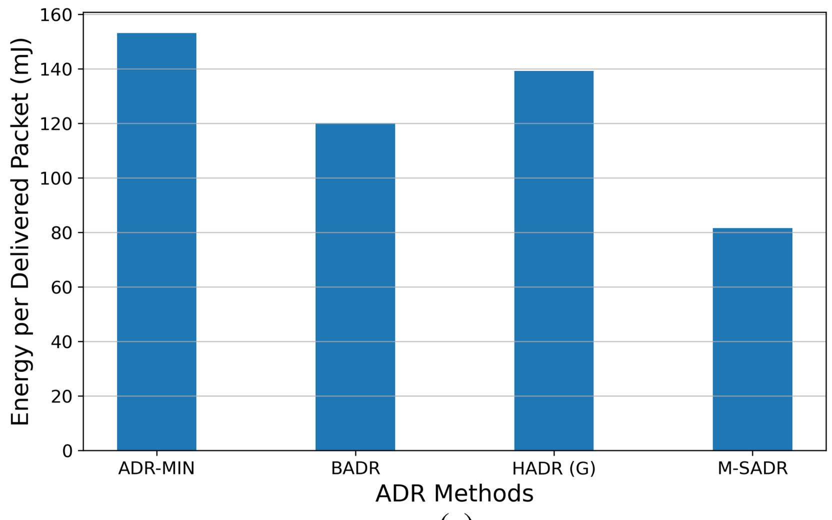
to assess the performance of the ADR methods under various conditions. The mobile device followed a consistent path across all methods to ensure fair comparison, with carefully controlled movement and stops at locations with varying signal conditions, including obstructions and open spaces.

In the first experimental scenario, only the mobile end device was used to measure the effectiveness of the methods, with no other devices present. Because no other LoRa networks or competing radio frequency signals were present in the area, there were no collisions from external transmissions. Any packet loss observed was due to interference from obstacles or the distance from the gateway. In the second scenario, the six stationary end devices were placed within the experimental area, continuously sending packets, each with a specific SF. The placement of each device was such that it could easily send packets to the gateway using its assigned SF. The movement of the mobile end device was the same as described in the first scenario.

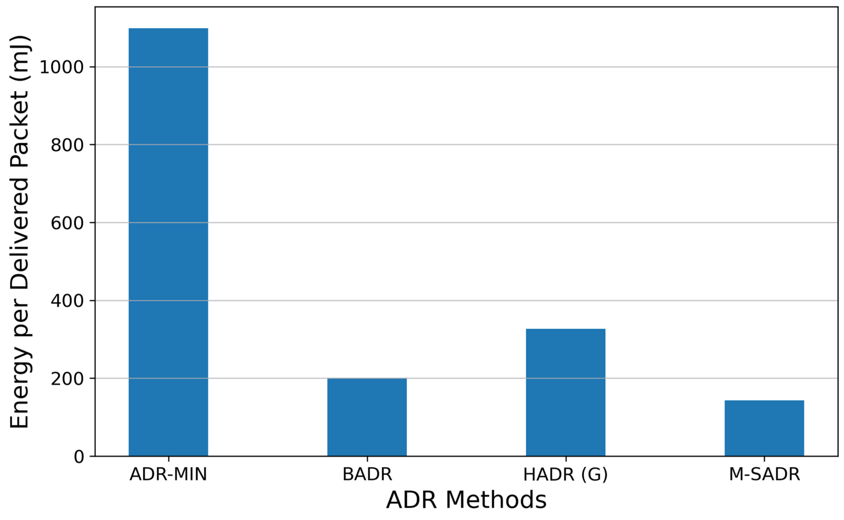
In both scenarios, the mobile end device transmitted packets periodically at random intervals, ranging from 8 to 12 s, to simulate typical network behavior under varying traffic conditions. In the second scenario, the six stationary end devices also transmitted packets at random intervals between 4 and 10 s. Additionally, as mentioned previously, all end devices were configured to send packets on a single channel, specifically in the 868.5 MHz band. This setup was designed to simulate a more congested environment, even with a limited number of devices, by creating packet traffic on the same channel.

The results for the PDR are shown in Figure 33. We can observe that ADR-MIN performs better than the other methods in low-traffic scenarios but underperforms in high-traffic conditions. On the other hand, M-SADR consistently delivers the best performance in terms of PDR across both low and high-traffic scenarios.

Figure 34a compares the average consumed ETP for the low traffic scenario, while Figure 34b compares the average consumed ETP for the high traffic scenario. Moreover, Figure 35a presents the average consumed EDP for the low traffic scenario, and Figure 35b for the high traffic scenario. The comparison indicates that in the implemented testbed, M-SADR achieves significantly lower energy consumption per successfully delivered



(a)



(b)

FIGURE 35 | Average consumed energy per delivered packet (EDP) in testbed: (a) low traffic and (b) high traffic scenario.

packet than its competitors for both low traffic and high traffic scenarios.

The results regarding the SF distribution are displayed in Figure 36. It appears that ADR-MIN cannot distribute packets across all SFs, as it selects the worst signal to make decisions, which leads to increased congestion in high-traffic scenarios. Additionally, BADR is limited to selecting only SF7, SF10, and SF12, which also becomes a drawback in high-traffic scenarios. Both HADR (G) and M-SADR achieve better SF distribution, leading to lower congestion in high traffic situations, as the gateway can simultaneously demultiplex packets with different SFs. Moreover, M-SADR performs better at lower SFs than HADR (G), which helps further reduce congestion in high-traffic scenarios.

Finally, Figure 37 presents the average latency of the delivered packets for all compared ADR methods in both low and high traffic scenarios. It appears that ADR-MIN significantly lags behind the other methods in terms of latency. In contrast, M-SADR demonstrates a notable improvement over the other methods, which can be attributed to its better selection of SFs, particularly the use of lower SFs. It is worth noting that HADR (G) does not perform well in terms of latency, as it includes an additional 8 bytes for GNSS information.

Overall, in the implemented testbed, M-SADR presents better results than its competitors across all evaluated metrics. Moreover, it appears to perform even better in real-world environments. This can be attributed to the fact that, although the simulation includes a path loss model accounting for factors like interference, obstacles, and shadowing effects, the shadowing component typically follows a log-normal distribution, which may influence the results. In contrast, in real-world scenarios–especially on a small scale–different propagation effects follow various statistical models, and their combined impact does not necessarily conform to a single predefined distribution.

6 | Discussion

The presented M-SADR method proves to be an effective solution for mobile LoRa end devices, maintaining a high PDR while keeping energy consumption lower than that of its competitors. Additionally, it performs well even at higher speeds.

Additionally, the M-SADR method is compatible with existing LoRaWAN v1.1 specifications and can be integrated into current deployments without any modifications. This is because all changes take place on the end-device side, with the only requirement for the network server being the deactivation of ADR. Specifically, on the network server side, the existing ADR mechanism is triggered every 20 packets or when a downlink packet with the ADR bit enabled is received. However, the ADR mechanism can be disabled on most

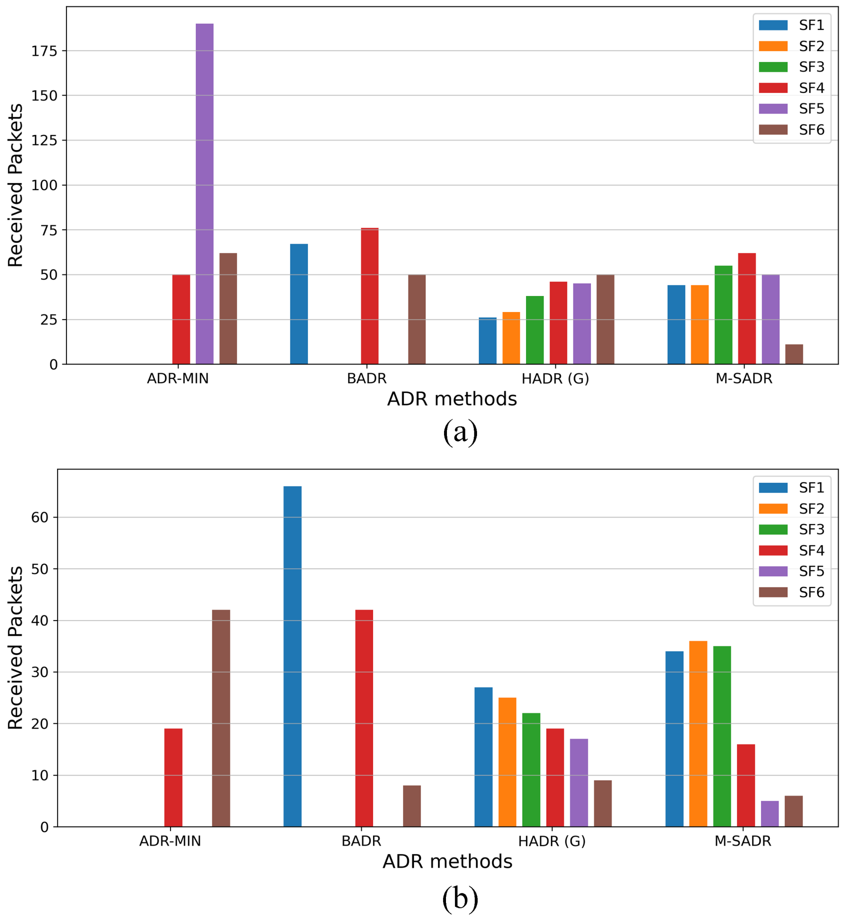
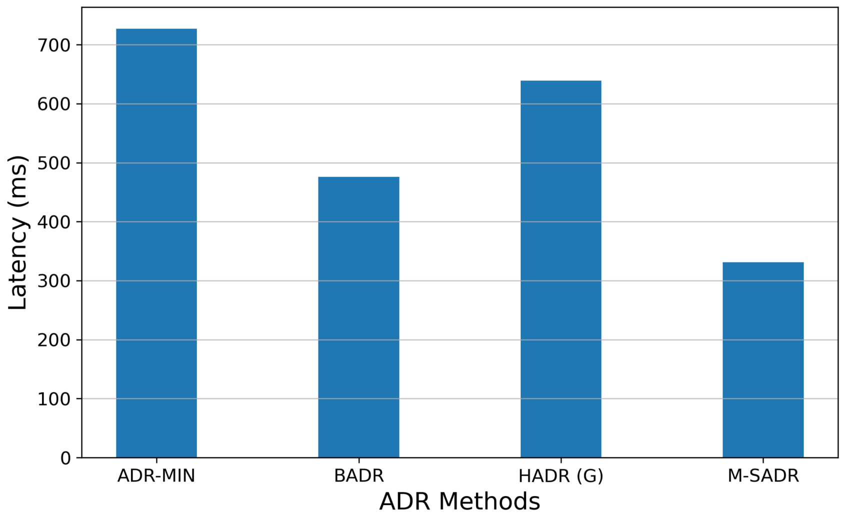
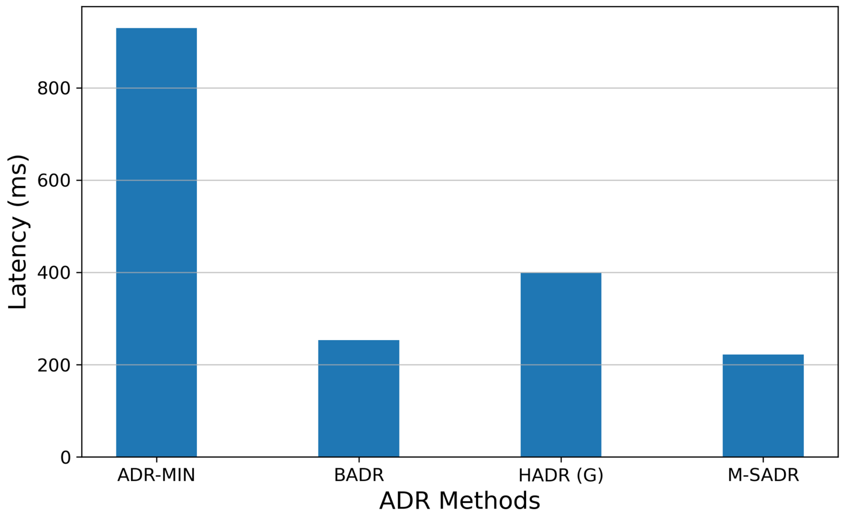


FIGURE 36 | Spreading factor distribution in testbed across ADR methods: (a) low traffic and (b) high traffic.



(a)



(b)

FIGURE 37 | Average latency for all ADR methods: (a) low traffic and (b) high traffic.

well-known network servers, such as The Things Network and ChirpStack. Furthermore, in the proposed M-SADR method, end devices do not send packets with the ADR bit enabled. Moreover, the M-SADR mechanism on the end-device side is significantly modified compared with the original ADR, but it retains the same packet structure, ensuring compatibility with current deployments.

Furthermore, the M-SADR mechanism, which operates on the end device side, is designed with low complexity while requiring minimal computational and memory resources, an essential factor for energy constrained end devices where efficiency is paramount. In many IoT applications, such as environmental monitoring, industrial automation, and smart agriculture, end devices operate on battery power for extended periods. As a result, optimizing computational efficiency and minimizing energy consumption are crucial to extending device lifespan and ensuring reliable performance. Additionally, to keep costs low, most LoRa end devices are designed with limited processing power and memory, making them unsuitable for handling complex algorithms or high computational loads.

By maintaining a lightweight design, M-SADR ensures seamless integration into existing LoRaWAN deployments without imposing additional hardware requirements, making it an ideal solution for resource-constrained environments.

Although M-SADR adjusts only the SF and TP, changes to other parameters such as CR and bandwidth are not expected to significantly impact its relative effectiveness, as these parameters influence all comparative ADR schemes in a similar manner. In our evaluation, we have already assessed the impact of varying CR and observed that while higher CR values increase packet length and the risk of collisions, this effect is consistent across all ADR strategies. Similarly, variations in bandwidth affect all approaches uniformly. While increasing bandwidth can reduce ToA and potentially lower the likelihood of collisions, it also decreases receiver sensitivity, which may degrade performance in environments with poor signal quality. However, as these effects apply consistently across ADR approaches, the relative performance benefits of M-SADR remain largely unaffected.

In addition, while this research evaluates the performance of M-SADR using a single transmission channel, utilizing all available channels is not expected to result in significant performance gains. This is because, with an equal distribution of packet transmissions across channels, the overall network load remains the same, even as the number of end devices increases. Thus, the performance would remain comparable in scenarios with more end devices, as long as the packet distribution across channels is uniform. However, to optimize network performance, future research could explore intelligent channel selection mechanisms, where end devices dynamically choose transmission channels based on factors such as signal strength, interference, and network congestion. This would help reduce packet collisions and improve overall throughput. Additionally, optimizing transmission timing could further enhance efficiency, allowing devices to adapt their transmission windows based on real-time network conditions and minimizing congestion. Notably, none of the evaluated ADR methods implement specific channel selection strategies, instead relying on the default random channel selection as specified by the LoRaWAN standard. By incorporating adaptive or prioritized channel selection techniques alongside optimized transmission timing, future versions of M-SADR could provide even better performance, particularly in dense networks with higher device counts.

Furthermore, the proposed M-SADR mechanism may encounter challenges in scenarios where transmissions are infrequent. For example, in this study, we examine the proposed algorithm in the context of continuous packet transmission with short intervals between consecutive packets. However, many scenarios involving LoRa end devices do not require frequent transmissions. Instead, packets are often sent at intervals of 15 to 20 minutes, or even longer. In such cases, mobile end devices may have moved significantly because their last transmission, which could impact the effectiveness of the M-SADR mechanism. The device's location and channel conditions may have changed considerably, making it more difficult for the algorithm to maintain optimal performance. To address these challenges, future research could explore alternative strategies, such as adaptive retransmission methods, which could re-evaluate transmission parameters based on device movement and environmental changes. These approaches could help ensure that M-SADR remains effective in scenarios with less frequent transmissions, where mobility and environmental factors play a more prominent role.

Additionally, continuous packet acknowledgment represents a potential limiting factor for the performance of the M-SADR mechanism. Acknowledging each transmitted packet individually can lead to unnecessary delays and congestion, especially in environments with high transmission frequencies or heavy network traffic. To improve efficiency, one potential solution is to implement an aggregated feedback acknowledgment mechanism. Instead of acknowledging each packet individually, this mechanism would evaluate the success of packet delivery over a series of transmissions or within specific time intervals. By reducing the frequency of acknowledgments, this approach minimizes overhead and network congestion, while still ensuring reliable communication. Furthermore, such an aggregated acknowledgment scheme could significantly enhance the scalability of the network, enabling it to handle larger volumes of data without compromising performance. This would ultimately help maintain the quality of service in scenarios where network resources are constrained, ensuring optimal performance even under heavy load conditions. Future research could focus on developing adaptive acknowledgment strategies that vary based on traffic patterns, packet delivery success rates, or network congestion levels, further enhancing the overall efficiency of the system.

7 | Conclusions

In this paper, we propose the M-SADR algorithm, an innovative method designed to reduce energy consumption in mobile LoRa end devices. We compare M-SADR with three other ADR methods: ADR-MIN, BADR, and HADR. The proposed method was evaluated in two distinct scenarios: one involving three mobile gateways tracking a moving group of end devices and another involving a single static gateway with multiple mobile end devices moving arbitrarily around it. A small scale testbed was also implemented to evaluate the results in real world scenarios.

The results demonstrate that M-SADR improves the PDR and energy efficiency per successfully delivered packet across various end-device speeds, reaching up to 24 mps. More specifically, based on both simulation scenarios, M-SADR achieves better performance by 2.4% in PDR and 5.7% in power consumption per successfully delivered packet. Furthermore, the conducted testbed proves the effectiveness of the proposed method, showing significant performance against its competitors.

Additionally, a key aspect of our work is the innovative methodology introduced in the first simulation scenario, where we dynamically adjust the positions of three mobile gateways to follow the movement of a group of end devices, ensuring that they remain in close proximity to the end devices in real time. Our methodology relies on RSSI-based position estimation, a low-cost yet effective technique that enables accurate repositioning of the mobile gateways.

For future work, we plan to explore M-SADR in more complex scenarios involving additional gateways and longer transmission intervals. Enhancing robustness may involve direct retransmissions for greater reliability in infrequent transmissions. Moreover, reducing the frequency of acknowledgments may help minimize network traffic and improve overall efficiency. Further research could also focus on optimizing transmission timing and channel selection to prevent collisions and enhance throughput in dynamic environments.

Acknowledgments

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No. 101097122 (ACROSS). Disclaimer: Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Commission. Neither the European Union nor the European Commission can be held responsible for them.

Conflicts of Interest

The authors declare no conflicts of interest.

References

1. M. Al mojamed, “On the Use of LoRaWAN for Mobile Internet of Things: The Impact of Mobility,” *Applied System Innovation.* 5, no. 1 (2022): 5, https://doi.org/10.3390/asi5010005.

2. P. Ribeiro, G. Dias, and P. Pereira, “Transport Systems and Mobility for Smart Cities,” *Applied System Innovation* 4, no. 3 (2021), https://doi.org/10.3390/asi4030061.

3. R. Dintn, S. Garca, and M. Zorrilla, “Fleet Management Systems in Logistics 4.0 Era: A Real Time Distributed and Scalable Architectural Proposal,” *Procedia Computer Science* 217 (2023): 806–815.

4. V. Agarwal, S. Tapaswi, and P. Chanak, “A Survey on Path Planning Techniques for Mobile Sink in IoT-Enabled Wireless Sensor Networks,” *Wireless Personal Communications* 119, no. 1 (2021): 211–238.

5. X. Liu, “Atypical Hierarchical Routing Protocols for Wireless Sensor Networks: A Review,” *IEEE Sensors Journal* 15, no. 10 (2015): 5372–5383, https://doi.org/10.1109/JSEN.2015.2445796.

6. B. Bhushan and G. Sahoo, “E2SR2: An Acknowledgement-Based Mobile Sink Routing Protocol With Rechargeable Sensors for Wireless Sensor Networks,” *Wireless Networks* 25, no. 5 (2019): 2697–2721.

7. C. Tunca, S. Isik, M. Y. Donmez, and C. Ersoy, “Ring Routing: An Energy-Efficient Routing Protocol for Wireless Sensor Networks With a Mobile Sink,” *IEEE Transactions on Mobile Computing* 14, no. 9 (2015): 1947–1960, https://doi.org/10.1109/TMC.2014.2366776.

8. J. H. Shin, J. Kim, K. Park, and D. Park, “Railroad: Virtual Infrastructure for Data Dissemination in Wireless Sensor Networks,” In: PE-WASUN’05. Association for Computing Machinery, (2005), New York, NY, USA:168174.

9. S. M. S. Mohd Daud, M. Y. P. Mohd Yusof, C. C. Heo, et al., “Applications of Drone in Disaster Management: A Scoping Review,” *Science & Justice* 62, no. 1 (2022): 30–42, https://doi.org/10.1016/j.scijus.2021.11.002.

10. H. Liang, S. C. Lee, W. Bae, J. Kim, and S. Seo, “Towards UAVs in Construction: Advancements, Challenges, and Future Directions for Monitoring and Inspection,” *Drones.* 7, no. 3 (2023), https://doi.org/10.3390/drones7030202.

11. A. Mohajer, M. Bavaghar, and H. Farrokhi, “Mobility-Aware Load Balancing for Reliable Self-Organization Networks: Multi-Agent Deep Reinforcement Learning,” *Reliability Engineering and System Safety* 202 (2020): 107056, https://doi.org/10.1016/j.ress.2020.107056.

12. V. Moysiadis, D. Skoutas, C. Skianis, T. Lagkas, V. Argyriou, and P. Sarigiannidis, *Mobile Gateways for LoRa End-Devices: Support Connectivity on the Move* (IEEE, 2024).

13. A. Adeel, M. Gogate, S. Farooq, et al., *A Survey on the Role of Wireless Sensor Networks and IoT in Disaster Management* (Springer Singapore, 2019): 57–66.

14. “LoRaWAN Mobile Applications: Blind ADR,” (2019).

15. A. Farhad and J. Y. Pyun, “HADR: A Hybrid Adaptive Data Rate in LoRaWAN for Internet of Things,” *ICT Express* 8, no. 2 (2022): 283–289, https://doi.org/10.1016/j.icte.2021.12.013.

16. J. Babaki, M. Rasti, and S. K. Taskou, “Improved Configuration of Transmission Variables for LoRaWAN in High-Noise Channels,” (2020): 1-6

17. V. Moysiadis, P. Sarigiannidis, V. Vitsas, and A. Khelifi, “Smart Farming in Europe,” *Computer Science Review* 39 (2021): 100345, https://doi.org/10.1016/j.cosrev.2020.100345.

18. J. A. Brenes and G. Marín-Raventós, “When One Wireless Technology Is Not Enough: A Network Architecture for Precision Agriculture Using LoRa, Wi-Fi, and LTE,” in *Intelligent Sustainable Systems*, eds. A. K. Nagar, D. S. Jat, G. Marín-Raventós, and D. K. Mishra (Springer Nature Singapore, 2022): 103–112.

19. R. O. Andrade and S. G. Yoo, “A Comprehensive Study of the Use of LoRa in the Development of Smart Cities,” *Applied Sciences* 9, no. 22 (2019), https://doi.org/10.3390/app9224753.

20. S. Jain, M. Pradish, A. Paventhan, M. Saravanan, and A. Das, “Smart Energy Metering Using LPWAN IoT Technology,” in *ISGW 2017: Compendium of Technical Papers*, ed. et al. (Singapore: Springer Singapore, 2018): 19–28.

21. N. Nurelmadina, M. K. Hasan, I. Memon, et al., “A Systematic Review on Cognitive Radio in Low Power Wide Area Network for Industrial IoT Applications,” *Sustainability.* 13, no. 1 (2021), https://doi.org/10.3390/su13010338.

22. B. Bhushan and G. Sahoo, “Recent Advances in Attacks, Technical Challenges, Vulnerabilities and Their Countermeasures in Wireless Sensor Networks,” *Wireless Personal Communications* 98, no. 2 (2018): 2037–2077, https://doi.org/10.1007/s11277-017-4962-0.

23. M. Sharma, A. Tandon, S. Narayan, and B. Bhushan, *Classification and Analysis of Security Attacks in WSNs and IEEE 802.15.4 Standards: A Survey* (IEEE, 2017): 1–5.

24. A. Triantafyllou, P. Sarigiannidis, T. Lagkas, I. D. Moscholios, and A. Sarigiannidis, “Leveraging Fairness in LoRaWAN: A Novel Scheduling Scheme for Collision Avoidance,” *Computer Networks* 186 (2021): 107735, https://doi.org/10.1016/j.comnet.2020.107735.

25. T. Bouguera, J. F. Diouris, J. J. Chaillout, R. Jaouadi, and G. Andrieux, “Energy Consumption Model for Sensor Nodes Based on LoRa and LoRaWAN,” *Sensors* 18, no. 7 (2018), https://doi.org/10.3390/s18072104.

26. M. Slabicki, G. Premsankar, and M. Di Francesco, “Adaptive Configuration of LoRa Networks for Dense IoT Deployments,” (2018): 1–9.

27. B. Zholamanov, A. Bolatbek, A. Saymbetov, et al., “Enhanced Reinforcement Learning Algorithm Based-Transmission Parameter Selection for Optimization of Energy Consumption and Packet Delivery Ratio in LoRa Wireless Networks,” *Journal of Sensor and Actuator Networks* 13, no. 6 (2024), https://doi.org/10.3390/jsan13060089.

28. V. Moysiadis, T. Lagkas, V. Argyriou, A. Sarigiannidis, I. D. Moscholios, and P. Sarigiannidis, “Extending ADR Mechanism for LoRa Enabled Mobile End-Devices,” *Simulation Modelling Practice and Theory* 113 (2021): 102388, https://doi.org/10.1016/j.simpat.2021.102388.

29. A. Farhad, D. H. Kim, S. Subedi, and J. Y. Pyun, “Enhanced LoRaWAN Adaptive Data Rate for Mobile Internet of Things Devices,” *Sensors* 20, no. 22 (2020), https://doi.org/10.3390/s20226466.

30. M. A. Lodhi, L. Wang, A. Farhad, et al., “A Contextual Aware Enhanced LoRaWAN Adaptive Data Rate for Mobile IoT Applications,” *Computer Communications* 232 (2025): 108042, https://doi.org/10.1016/j.comcom.2024.108042.

31. D. H. Kim and J. Y. Pyun, “AI-Driven Adaptive Data Rate for LoRaWAN Location-Based Services,” *IEEE Access* 12 (2024): 168349–168359, https://doi.org/10.1109/ACCESS.2024.3494874.

32. M. Alenezi, K. K. Chai, A. S. Alam, Y. Chen, and S. Jimaa, “Unsupervised Learning Clustering and Dynamic Transmission Scheduling for Efficient Dense LoRaWAN Networks,” *IEEE Access* 8 (2020): 191495–191509, https://doi.org/10.1109/ACCESS.2020.3031974.

33. A. Gorrela and N. Choudhury, “A Clustering-Based Adaptive Data Rate Technique for Industrial LoRaWAN-IoT Networks,” *IEEE Internet of Things Journal* (2025): 1–1, https://doi.org/10.1109/JIOT.2024.3492233.

34. Z. Ali, K. N. Qureshi, A. S. Al-Shamayleh, A. Akhunzada, A. Raza, and M. F. U. Butt, “Delay Optimization in LoRaWAN by Employing Adaptive Scheduling Algorithm With Unsupervised Learning,” *IEEE*

Access 11 (2023): 2545–2556, https://doi.org/10.1109/ACCESS.2023.3234188.

35. J. Park, K. Park, H. Bae, and C. K. Kim, "EARN: Enhanced ADR With Coding Rate Adaptation in LoRaWAN," *IEEE Internet of Things Journal* 7, no. 12 (2020): 11873–11883, https://doi.org/10.1109/JIOT.2020.3005881.

36. D. Y. Kim, S. Kim, H. Hassan, and J. H. Park, "Adaptive Data Rate Control in Low Power Wide Area Networks for Long Range IoT Services," *Journal of Computational Science* 22 (2017): 171–178, https://doi.org/10.1016/j.jocs.2017.04.014.

37. N. Benkahla, H. Tounsi, Y. Q. Song, and M. Frikha, *Enhanced ADR for LoRaWAN Networks With Mobility* (IEEE, 2019): 1–6.

38. H. Wang, X. Zhang, J. Liao, Y. Zhang, and H. Li, "An Improved Adaptive Data Rate Algorithm of LoRaWAN for Agricultural Mobile Sensor Nodes," *Computers and Electronics in Agriculture* 219 (2024): 108773, https://doi.org/10.1016/j.compag.2024.108773.

39. C. Chen, J. Luo, Z. Xu, R. Xiong, D. Shen, and Z. Yin, "Enabling Large-Scale Low-Power LoRa Data Transmission via Multiple Mobile LoRa Gateways," *Computer Networks* 237 (2023): 110083, https://doi.org/10.1016/j.comnet.2023.110083.

40. S. Sobhi, A. Elzanaty, M. Y. Selim, A. M. Ghuniem, and M. F. Abdelkader, "Mobility of LoRaWAN Gateways for Efficient Environmental Monitoring in Pristine Sites," *Sensors* 23, no. 3 (2023), https://doi.org/10.3390/s23031698.

41. W. Choi, Y. S. Chang, Y. Jung, and J. Song, "Low-Power LoRa Signal-Based Outdoor Positioning Using Fingerprint Algorithm," *ISPRS International Journal of Geo-Information* 7, no. 11 (2018), https://doi.org/10.3390/ijgi7110440.

42. V. Delafontaine, F. Schiano, G. Cocco, A. Rusu, and D. Floreano, "Drone-Aided Localization in LoRa IoT Networks," (2020): 286-292

43. I. Daramouskas, D. Mitroulias, I. Perikos, M. Paraskevas, and V. Kapoulas, *Localization in LoRa Networks Based on Time Difference of Arrival* (Springer International Publishing, 2022): 130–143.

44. M. Anjum, M. A. Khan, S. Ali Hassan, A. Mahmood, and M. Gidlund, "Analysis of RSSI Fingerprinting in LoRa Networks," (2019): 1178-1183.

45. M. Anjum, M. A. Khan, S. A. Hassan, A. Mahmood, H. K. Qureshi, and M. Gidlund, "RSSI Fingerprinting-Based Localization Using Machine Learning in LoRa Networks," *IEEE Internet of Things Magazine* 3, no. 4 (2020): 53–59, https://doi.org/10.1109/IOTM.0001.2000019.

46. A. Vazquez-Rodas, F. Astudillo-Salinas, C. Sanchez, B. Arpi, and L. I. Minchala, "Experimental Evaluation of RSSI-Based Positioning System With Low-Cost LoRa Devices," *Ad Hoc Networks* 105 (2020): 102168, https://doi.org/10.1016/j.adhoc.2020.102168.

47. K. Z. Islam, D. Murray, D. Diepeveen, M. G. Jones, and F. Sohel, "LoRa-Based Outdoor Localization and Tracking Using Unsupervised Symbolization," *Internet of Things* 25 (2024): 101016, https://doi.org/10.1016/j.iot.2023.101016.

48. S. Kaven, L. Bornholdt, and V. Skwarek, "Authentication by RSSI-Position Based Localization in a LoRa LPWAN," in (2020): 448–454.

49. G. Li, E. Geng, Z. Ye, Y. Xu, J. Lin, and Y. Pang, "Indoor Positioning Algorithm Based on the Improved RSSI Distance Model," *Sensors* 18, no. 9 (2018), https://doi.org/10.3390/s18092820.

50. M. Maduranga, V. Tilwari, and R. Abeysekera, "Improved-RSSI-Based Indoor Localization by Using Pseudo-Linear Solution With Machine Learning Algorithms," *Journal of Electrical Systems and Information Technology* 11, no. 1 (2024): 10.

51. A. Waqar, I. Ahmad, D. Habibi, and Q. V. Phung, "A Range Error Reduction Technique for Positioning Applications in Sports," *Journal of Engineering* 2021, no. 2 (2021): 73–84.

52. HARDWARIO, "Open LoRaWAN Modem for Murata Type ABZ Module. GitHub repository," (2025).