Mobile Robotics for Throughput Testing in a Reconfigurable Test Environment

Eduardo Araújo
Federal University of Pernambuco
Recife, Brazil
ewsa@cin.ufpe.br

Breno Miranda
Federal University of Pernambuco
Recife, Brazil
bafm@cin.ufpe.br

ABSTRACT

The growing demand for wireless connectivity in applications such as the Internet of Things (IoT), autonomous vehicles, and industrial automation requires accurate performance evaluation of networks in mobile and dynamic environments. This work proposes the development of a reconfigurable test environment based on mobile robots, using Raspberry Pi Zero 2 W boards to periodically collect Wi-Fi signal data from different positions within a physical space. Two experimental scenarios are considered: (i) client mobility, in which a Mesh router is mounted on a robot that performs measurements while moving, while the Raspberry Pi remains stationary collecting data; and (ii) infrastructure mobility, in which Mesh routers mounted on robots move while the Raspberry Pi remains fixed, collecting data. Data collection is performed using native commands from the embedded Linux system, and results are stored in JSON files for later analysis. This approach directly evaluates signal degradation, connection instability, handover events, physical obstacles, and network fluctuations in realistic scenarios. The expected outcomes of our research include the creation of a valuable dataset for validating wireless connectivity quality and supporting the development of more robust solutions for mobile networks and dynamic environments. This work presents an innovative reconfigurable testbed combining mobile robots and automated diagnostics capable of accurately and reproducibly evaluating handover and latency in Wi-Fi Mesh networks.

KEYWORDS

Reconfigurable Testbed, Mobile Robotics, Wi-Fi Networks, Raspberry Pi, Performance Evaluation, RSSI, IoT

1 Introduction

Consider an autonomous cleaning robot operating inside a hospital or a factory floor. As it moves between rooms, Wi-Fi signal quality fluctuates, latency spikes may occur, and connection handovers become inevitable. Traditional network testbeds (typically composed of static nodes) are not equipped to simulate or evaluate this type of mobile behavior accurately. As a result, developers often discover connectivity-related issues only after deploying systems in the real world.

To address this limitation, this work proposes a reconfigurable test environment that leverages mobile robotics to evaluate wireless network performance in realistic mobility scenarios. The testbed uses Raspberry Pi-based units and Wi-Fi Mesh routers to measure connectivity parameters under controlled movement conditions which adopts a modular architecture composed of mobile robots, fixed sensing units, and a mesh-based wireless infrastructure. Each unit plays a specific role in the experiment, as detailed in Section 4.

The Arrange–Act–Assert is a methodological model which pattern provides a natural and intuitive flow for creating a unit test case[20]. The system enables structured and repeatable experiments which robots move based on signal strength heuristics, and diagnostic modules automatically identify connectivity degradation, latency spikes, and handover events.

In preliminary experiments, the system successfully identified zones of signal instability, handover occurrences, and latency spikes during real mobility. These findings validate the testbed's capability to reproduce critical connectivity issues under dynamic conditions.

This work presents the design and validation of a modular and reconfigurable testbed using mobile robotics to evaluate wireless network performance under realistic mobility conditions.

The key contributions of this work are:

- A mobile and reconfigurable testbed architecture combining autonomous robots (MUUTs), static sensing units (FUUTs), and a mesh-based network backbone (SUs).
- The integration of the Arrange–Act–Assert methodology to enable repeatable, structured experiments under real movement conditions.
- A diagnostic layer that performs real-time analysis of connectivity metrics, detecting latency variations, signal degradation, and handover events during motion.

The remainder of this paper is organized as follows. Section 2 provides an overview of related work, while Section 3 outlines the evaluation architecture and methodology used in our study. Section 4 presents our preliminary results, followed by Section 5, which explores the practical contributions of our proposed approach. Finally, Section 6 summarizes our key findings and concludes the paper.

2 Related Work

Testbeds and Network Mobility Evaluation

Several wireless testbeds have supported experimental validation in static environments. GENI [18] and Emulab [10] provide structured and reproducible environments for testing protocols and topologies, but lack physical mobility and real-time wireless behavior under motion [8]. These systems assume fixed infrastructure, which limits their ability to simulate handover instability, fluctuating signal strength, or spatial interference.

More recent efforts have explored mobility-aware scenarios. Zou et al. [22] developed an indoor Wi-Fi mapping robot using Raspberry Pi, collecting spatial signal data. However, their work is passive and limited to data collection without structured experimental flow or active performance evaluation. Zhou et al. [12] applied RSSI (Received Signal Strength Indication) combined with machine learning for indoor localization, but relied entirely on static sensors, which prevents testing the impact of motion on connectivity dynamics, such as handovers or latency spikes.

Efforts to improve mesh network performance have focused on protocol-level optimization. Gokalgandhi et al. [9] introduced a routing metric that balances reliability and latency, achieving superior results in simulation. Similarly, Zhang et al. [21] proposed a ticket-based authentication scheme for fast handover, reducing latency and communication overhead. Li et al. [14] compared their approach to IEEE 802.11 authentication and demonstrated significant gains. However, all these studies validated their methods in fixed, controlled environments (either simulations or non-mobile setups) which prevents evaluating real-world effects such as overlapping coverage zones or environmental signal interference. While these proposals are valuable at the protocol layer, the lack of empirical validation in physical, mobile scenarios raises questions about their effectiveness under real-world dynamics.

In contrast, our work provides a physical testbed that incorporates mobility, enabling the empirical validation of these approaches under dynamic topologies. By simulating realistic motion patterns and collecting live metrics such as latency, handover events, and RSSI fluctuations, our system bridges the gap between protocol design and field behavior.

Kunze et al. [13] address the challenges developers face when testing mobile robots using simulation-based scenarios. They propose a composable and executable approach for modeling test environments and task specifications, enabling developers to define reusable, dynamic scenarios that adapt to different testing goals. While their work focuses on virtual environments and simulation frameworks, it highlights the importance of modularity, automation, and repeatability in robotics testing. Our work builds upon these principles by extending them into a physical testbed that not only includes mobility and modular architecture but also introduces a diagnostic layer that performs real-time analysis of connectivity metrics, such as latency variation, signal degradation, and handover detection, enabling empirical evaluation of wireless behavior under motion.

Mobile Robotics in Testing Infrastructures

Robotics has been applied to automate mobile testing, primarily targeting user interaction [15]. The R-DLD system [7] replicated device orientation to detect data loss, while Mao et al. [16] and Qian et al. [19] developed robotic testers for black-box UI validation.

In contrast, our work focuses on network conditions and connectivity stability. Robotic motion is used to induce natural network variation and stress handover mechanisms. We also incorporate structured experimentation (Arrange–Act–Assert), real-time diagnostics, and system feedback, elements not present in previous robotics-based testbeds. To the best of our knowledge, this is the first platform to combine robotic mobility, mesh network interaction, automated diagnostics, and reproducibility in a unified setup for wireless performance testing.

Software-Oriented Perspectives on Testbeds

In Software Engineering, testbeds are essential for validating application behavior under varied execution conditions. Anka et al. [1] proposed combining static and dynamic models to improve the realism of mobile app testing, reinforcing the importance of dynamic environments in structured validation.

The Arrange–Act–Assert pattern, widely adopted in software testing [17], has been adapted for network evaluation [3], though not incorporating physical mobility. Similarly, distributed systems often rely on simulation-based strategies [5], which fail to capture the complexity of real-world wireless behavior.

Our work addresses this gap by integrating physical mobility into a testbed grounded in software testing principles. This enables evaluation in scenarios where connectivity and performance fluctuate, a context often overlooked by traditional tools.

Table 1: Comparison of physical mobility, automatic diagnosis, modularity, and software testing support across related testbeds and the proposed system

Testbed	Physical Mobility	Automatic Diagnosis	Modularity	Software Testing Support
GENI [18]	No	No	Yes	Limited
Emulab [10]	No	No	Yes	Limited
Zou et al. [22]	Yes	No	No	No
Kotaru et al. [12]	No	Yes	No	No
Engel et al. [5]	No	Yes	Yes	Yes
This work	**Yes**	**Yes**	**Yes**	**Yes**

As shown in Table 1, none of the existing testbeds simultaneously support physical mobility, automated diagnostics, modularity, and structured software testing. Our platform, to the best of our knowledge, is the first to unify these aspects, enabling reproducible and realistic wireless performance evaluation.

3 Evaluation Architecture and Methodology

This section presents the architecture of the test environment, including the organization of units, the experimental model, and technologies used.

Organization of Units

The testbed comprises three unit types: **MUUT** (Mobile Unit Under Tasking): mobile robots with embedded Wi-Fi Mesh routers. This type of unit collect signal metrics while moving, guided by signal quality. **FUUT** (Fixed Unit Under Tasking): stationary Raspberry Pi devices positioned to collect parallel data as static sensors. **SU** (Support Unit): fixed 2.4 GHz Wi-Fi Mesh routers, acting as connectivity infrastructure and data sinks via SSH.

To isolate signal effects from environmental interference, tests were conducted in obstacle-free indoor settings, where variation stems solely from distance and topology [12].

Arrange–Act–Assert Model

The evaluation follow the Arrange–Act–Assert testing pattern. *Arrange*: distribution of MUUTs, FUUTs, and SUs in the environment, with defined starting points, collection intervals, and latency thresholds. *Act*: MUUTs run network scans (`iwlist`, `ping`) and move based on signal quality, collecting data every 30 seconds. *Assert*: data is sent via SSH and analyzed to detect `handover` and packet loss from JSON logs. This approach ensures reproducibility, comparability between runs, and controlled performance analysis under mobility.

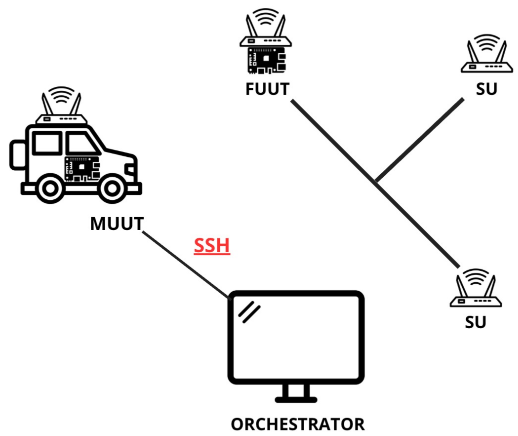


Figure 1: Overview of the proposed system architecture showing the roles of MUUT, FUUT, and SU in the physical testbed.

Technologies Used

The environment integrates embedded hardware, Linux tools, and Python scripts. The system uses native Linux commands such as iwlist (for Wi-Fi scanning) and ping (for latency measurement) to collect connectivity data during experiments. Key components include: **Hardware**: Raspberry Pi Zero 2 W (MUUTs/FUUTs), modified vacuum robots, and Wi-Fi Mesh routers. **OS**: Raspberry Pi OS (64-bit). **Data tools**: `iwlist`, `ping`, `ifconfig`. **Automation**: Python 3.9+ with `subprocess`, `json`, `pandas`, `matplotlib`. **Execution**: `systemd` automates startup on MUUTs. **Transmission**: data is sent via SSH to a central node.

Tests ran indoors (1–10 meters), simulating mobile device behavior through periodic data collection and MUUT movement.

Collection and Movement Strategy

Each MUUT periodically scans networks using `iwlist scan` and `ping`. Robots move one at a time to avoid interference, guided by a heuristic that favors stronger RSSI values and lower latency.

As the environment has no physical obstacles, signal variation reflects only distance and topology effects [12]. Robot positions are updated dynamically based on RSSI readings at each step.

Evaluation Description, Data Transmission and Storage

Three scenarios were designed to evaluate system scalability and robustness, grouped into two categories: **Client Mobility** (Scenario 1): 1 moving MUUT and 2 static FUUTs. **Infrastructure Mobility** (Scenarios 2 and 3): Scenario 2: 2 MUUTs and 1 FUUT. Scenario 3: 3 MUUTs with no FUUTs.

All evaluations took place indoors, with devices spaced 1–10 meters apart. Each MUUT collected data every 30 seconds for 10 minutes, capturing RSSI, latency, packet loss, and handover events. After each mission, MUUTs transmit locally stored JSON logs via SSH to a central SU for analysis. This offloading strategy avoids congestion and enables centralized post-processing for visualization and pattern detection.

Automated Performance Diagnostics

The system includes the `MeshAnalyzer` module, which performs continuous diagnostics based on configurable parameters:

Latency threshold: >100 ms triggers high-latency alerts. **Packet loss threshold**: >5% triggers interference warnings. **Correlation**: simultaneous high latency and loss = critical event. **Time window**: 5-minute sliding window (adjustable).

It also generates real-time reports with: Average/peak latency; Average packet loss; Critical event count and Recommended actions (e.g., "check link quality").

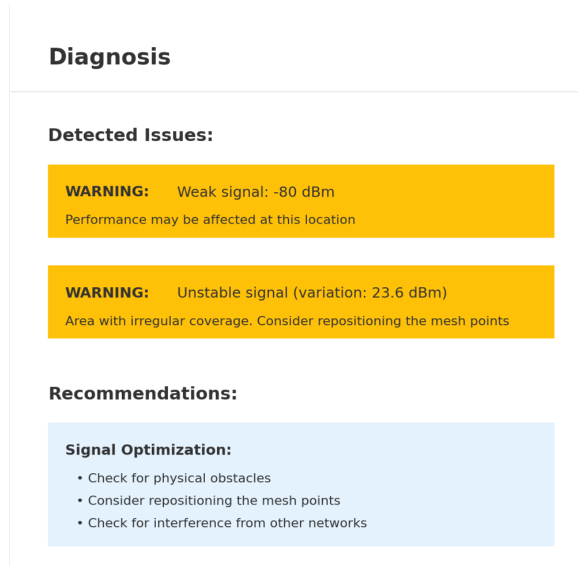


Figure 2: Example of automatic diagnostics with weak signal alerts and system-generated recommendations.

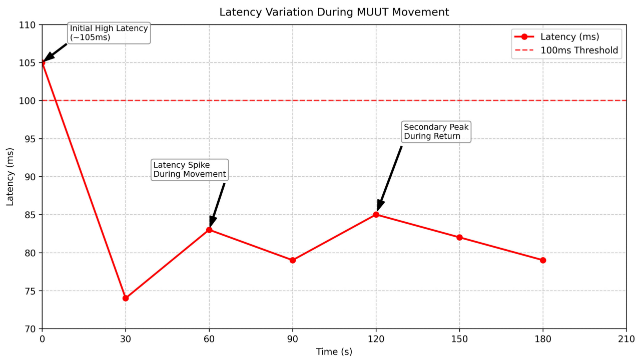
These features enable real-time performance monitoring and adaptive feedback, as shown in Figure 2.

4 Results and Analysis

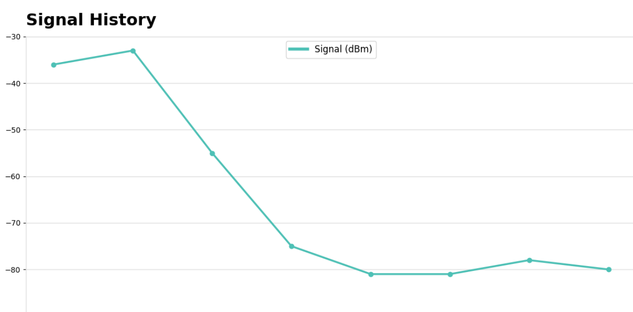
This section presents the evaluation results across three scenarios, focusing on: (i) automatic recommendations, (ii) log analysis, (iii) emerging connectivity patterns, and (iv) system scalability. Metrics analyzed include latency, handover behavior, packet loss, and the ability to operate with multiple concurrent MUUTs.

System-Generated Recommendations

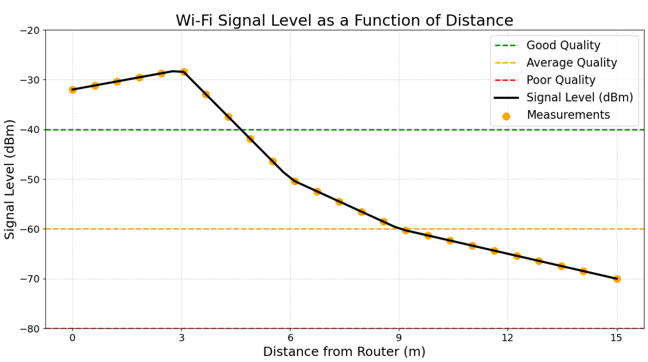
The diagnostic module generated automatic recommendations based on thresholds. Table 2 summarizes the main detected conditions and suggested actions. These alerts demonstrate the potential for future autonomous adaptation, such as rerouting or repositioning based on performance degradation. Latency $\leq$ 100 ms was considered acceptable, while values above 100 ms were flagged as high-latency events, aligned with FCC and industry guidelines [6, 11]. Figure 3a shows examples of such spikes during MUUT movement.



(a) Latency variation during MUUT movement, showing spikes above the 100 ms threshold



(b) Signal strength history (RSSI) as MUUT moves away from mesh access points



(c) Wi-Fi signal variation by room over time, highlighting areas of weak coverage

Figure 3: Evaluation results

Log Analysis

Each MUUT stores logs in JSON format, including `SSID`, `BSSID`, `RSSI`, `latency_ms`, `packet_loss`, and `handover`. Listing 1 shows an example of a log.

This log captures a handover event to a stronger `BSSID`, common in areas with degraded signal and latency peaks over 150 ms [2].

As shown in Figure 3b, the RSSI drops from −40 dBm to below −80 dBm as the MUUT moves away from the mesh APs.

Table 2: Triggers and recommended actions by the diagnostic system

Detected Condition	Recommended Action
Latency > 100 ms	Investigate causes of delay; reposition MUUT
Packet loss > 5%	Check for interference, congested channels, or physical obstacles
High latency + loss	Redirect MUUT and adjust mesh configuration on SU
RSSI < -75 dBm	Reduce distance to SU or redesign layout


```
{
  "timestamp": "2025-02-18 15:26:41",
  "SSID": "wanderley",
  "BSSID": "42:ED:00:28:11:EE",
  "Signal Level (dBm)": -47,
  "latency_ms": 77.216,
  "packet_loss": false,
  "handover": true,
  "new_bssid": "42:ED:00:28:11:EE"
}
```


Listing 1: Example of a log

Observed Emerging Patterns

Data analysis revealed unstable signal zones, mainly at the edges of SU coverage. These areas showed latency >150 ms, packet loss, and frequent handovers, forming degradation patterns linked to spatial positioning. The degradation patterns by room are illustrated in Figure 3c.

Scalability Evaluation

With up to three concurrent MUUTs, the system maintained stable data collection. However, shared SU zones showed increased latency, packet collisions, and occasional loss, expected under traffic saturation [3].

Despite this, all `systemd`-managed services remained operational, and data transmission to the SU succeeded post-mission. These results validate the system's robustness for small-scale distributed mobility scenarios.

Table 3: Summary of performance metrics observed in each experimental scenario

Scenario	Avg Latency (ms)	Handovers	Avg RSSI (dBm)	Packet Loss Rate
Scenario 1	85.2	2	-58.7	1.2%
Scenario 2	97.5	4	-62.4	2.3%
Scenario 3	112.8	6	-70.1	5.8%

Interface and Automated Diagnostics

The platform includes a real-time web interface (Flask + Socket.IO) for visualizing network metrics from MUUT and FUUT units. In addition to automated data collection, it supports live monitoring of signal strength, latency, handovers, and alerts, enabling dynamic diagnostics and informed decisions during experiments.

Real-Time Monitoring Interface The dashboard (Figure 4) displays real-time metrics: SSID, BSSID, RSSI, latency, signal/battery history, and diagnostic alerts. Alerts trigger dynamically when thresholds

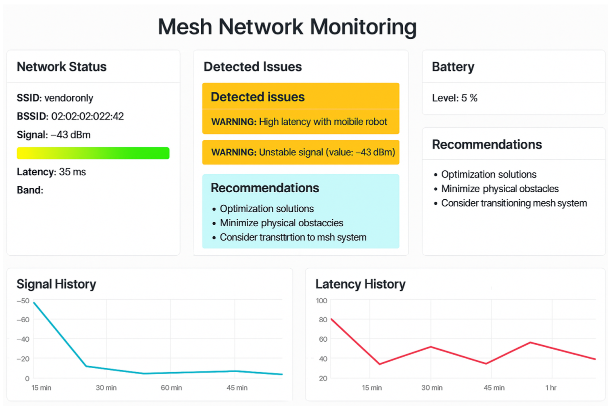


Figure 4: Web interface showing real-time Wi-Fi metrics and automated diagnostics.

are exceeded (e.g., RSSI < -75 dBm or latency > 100 ms), enabling users to respond to network degradation and mobility impact in real time.

Automated Diagnostic Cycle Beyond graphical visualization, the system incorporates an automated diagnostic cycle, illustrated in Figure 5. This cycle executes periodic evaluations based on logical rules, enabling real-time performance analysis and proactive issue detection. The decision logic includes:

- `if latency > 100 ms`: append a recommendation to investigate latency causes;
- `if avg_packet_loss > 5%`: trigger an alert indicating potential interference or signal degradation;
- `if latency high && packet loss high`: classify the condition as a critical connectivity event.

During each evaluation cycle, diagnostics are updated and context-aware recommendations are generated, suggesting adjustments such as repositioning MUUTs or modifying SU configurations. This logic enables responsive and adaptive behavior aligned with real-time network dynamics.

The automated cycle strengthens the testbed's capability to operate autonomously, delivering actionable feedback that enhances the precision and efficiency of mobility-based performance testing.

5 Discussion and Contributions

This section discusses the innovative aspects of the proposed approach, the practical contributions to the advancement of mobility-enabled testbeds, and the limitations encountered during our evaluation.

The absence of physical obstacles in our experimental setup is critical to isolating the effects of mobility on network performance. In similar environments, the lack of obstacles has been recognized as crucial for evaluating network behavior, particularly when assessing signal degradation and the effects of mobility [12]. This controlled setup ensures that the observed variations in network performance can be attributed to the movement and topology changes, rather than environmental interference, providing a clearer understanding of the dynamics involved.

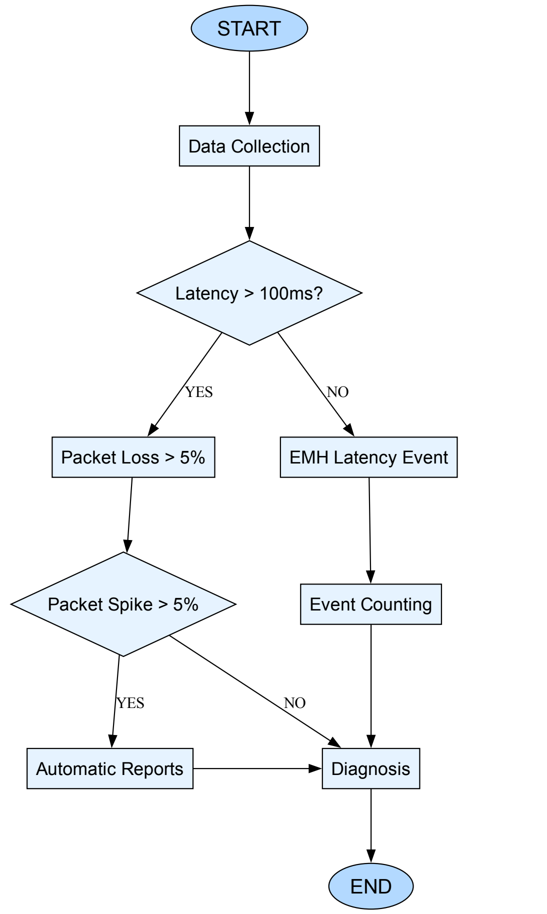


Figure 5: Decision flow of the automated diagnostic system, based on latency and packet loss.

Methodological Innovation

Leveraging mobile robotics as the basis for wireless network experimentation marks a significant departure from traditional static testbeds. The proposed system introduces a modular architecture that distinctly separates mobile units (MUUT), fixed units (FUUT), and support units (SU), enabling flexible scenario configuration and the targeted evaluation of variables such as mobility, topology, and coverage.

The use of the Arrange–Act–Assert model adds methodological rigor by structuring each experiment into reproducible phases: setup (Arrange), execution (Act), and validation (Assert). This structure facilitates hypothesis-driven experimentation and consistent analysis across runs.

Additionally, the integration of accessible and low-cost technologies, such as Raspberry Pi boards, Mesh routers, and Python-based automation, demonstrates the feasibility and replicability of the approach for both academic and applied research contexts.

Unlike prior mobile mapping robots or testbeds that only collect signal data, our system includes a continuous diagnostic engine, structured testing model, and modular expansion capabilities.

Limitations and Challenges

Despite its methodological contributions, this work presents limitations that impact the generalization of results. The movement of the MUUTs was restricted to an indoor environment without physical obstacles, limiting the realism of more complex urban or industrial scenarios. Additionally, the system operates exclusively on the 2.4 GHz band and does not consider multiple channels or cross-band interference, which are common in practical deployments.

A critical limitation is the absence of positioning sensors (such as GPS or IMU), which prevents a direct correlation between physical location and signal strength. As a result, robot movement is guided solely by RSSI variations, which can introduce noise and ambiguity in handover decisions.

Furthermore, data transmission occurs only at the end of each mission, with no real-time communication between MUUTs and SU during data collection. This reduces the system's capacity to react dynamically to critical events during execution.

Nonetheless, the primary contributions of this work include: (i) the structured use of mobile robotics within a reconfigurable wireless testing environment; (ii) the definition of a modular architecture with well-defined unit roles (MUUT, FUUT, SU); and (iii) the automation of performance diagnostics through continuous data collection and adaptive analysis logic. While some prior works explore mobile robotics for signal mapping, to the best of our knowledge, this is the first approach to combine mobility, automated diagnostics, and experimental formalization in a modular testbed designed for evaluating dynamic wireless networks.

6 Conclusion and Future Work

This work presents the development of a reconfigurable test environment based on mobile robotics for evaluating Wi-Fi networks in realistic mobility scenarios. The approach proved effective in identifying instability zones, performing autonomous handovers, and simulating overload scenarios with multiple mobile units. The proposed architecture features three distinct types of units (MUUT, FUUT, and SU), which operate in a coordinated manner to enable automated collection of key metrics such as RSSI, latency, and handover events. The adoption of the Arrange–Act–Assert model allows for the systematic execution of experiments and the structured analysis of results.

As future work, we plan the following developments: Expand the scale to support more than 10 MUUTs, enabling simultaneous movement and coordination among robots; Integrate sensors such as GPS, IMU, and compass for more precise spatial analysis; Add real-time communication with the SU, allowing for adaptive responses during data collection; Apply machine learning algorithms to predict critical zones and optimize movement routes and investigate energy management strategies for MUUT devices, inspired by tools such as the one presented by Kamble et al. [4], to enhance device autonomy in long-duration missions.

These improvements aim to transform the proposed environment into a robust platform for autonomous, adaptive, and scalable testing, contributing to research in wireless networks, edge computing, and distributed systems with mobility.

ACKNOWLEDGMENTS

This work was partially supported by a grant from the National Council for Scientific and Technological Development (Grant CNPq-Universal 408651/2023-7).

REFERENCES

[1] Salihu Anka, Rosziati Ibrahim, Bestoun Ahmed, Kamal Zamli, and Asmau Usman. 2019. AMOGA: A Static-Dynamic Model Generation Strategy for Mobile Apps Testing. *IEEE Access* PP (01 2019). https://doi.org/10.1109/ACCESS.2019.2895504

[2] Balint Bicski and Adrian Pekar. 2024. Unveiling Latency-Induced Service Degradation: A Methodological Approach With Dataset. *IEEE Access* 12 (2024), 128097–128115. https://doi.org/10.1109/ACCESS.2024.3456588

[3] K. Mani Chandy and J. Misra. 1985. Distributed Snapshots: Determining Global States of Distributed Systems. *ACM Transactions on Computer Systems* 3, 1 (1985), 63–75. https://doi.org/10.1145/214451.214456

[4] Aastha Chawla, Sreya Goyalia, and Nidusha Kannan. 2022. EMT: Software-based Energy Management Tool for WiFi IoT Devices. (2022).

[5] Fábio Engel de Camargo and Elias P. Duarte Jr. 2022. Um Simulador para Redes Sem Fio sob o Modelo SINR. *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)* (2022), 1–10. https://github.com/fabioengel/SINR

[6] Federal Communications Commission. 2021. Measuring Broadband America - A Report on Latency. Online. https://www.fcc.gov/reports-research/reports/measuring-broadband-america/measuring-fixed-broadband-report-2021.

[7] Davi Freitas, Breno Miranda, and Juliano Iyoda. 2024. Robotic-supported Data Loss Detection in Android Applications. In *Anais do XXXVIII Simpósio Brasileiro de Engenharia de Software* (Curitiba/PR). SBC, Porto Alegre, RS, Brasil, 1–11. https://doi.org/10.5753/sbes.2024.2899

[8] Jorge R Beingolea Garay, Roberto K Hiramatsu, Sergio T Kofuji, and Roseli de Deus Lopes. 2008. Avaliação experimental de redes mesh com os classmate pcs da Intel. *JISTEM-Journal of Information Systems and Technology Management* 5, 2 (2008), 235–250.

[9] Bhargav Gokalgandhi et al. 2022. Reliable Low-Latency Wi-Fi Mesh Networks. *IEEE Internet of Things Journal* (2022).

[10] Nikhil Handigol, Brandon Heller, Vimal Kumar Jeyakumar, David Mazières, and Nick McKeown. 2012. Reproducible Network Experiments Using Container-based Emulation. In *Proc. 8th Int. Conf. on Emerging Networking Experiments and Technologies (CoNEXT)*. ACM, Nice, France, 253–264.

[11] Intel Corporation. 2023. Wireless Latency and Performance Guidelines. Online. https://www.intel.com/content/www/us/en/tech-tips/wireless-latency-and-performance.html.

[12] Manikanta Kotaru, Kiran Joshi, Dinesh Bharadia, and Sachin Katti. 2015. SpotFi: Decimeter Level Localization Using WiFi. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX Association, 269–282. https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/kotaru

[13] Lars Kunze, Mihai Emanuel Dolha, and Michael Beetz. 2011. Logic programming with simulation-based temporal projection for everyday robot object manipulation. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 3238–3244. https://doi.org/10.1109/IROS.2011.6094996

[14] Celia Li et al. 2013. Efficient authentication for fast handover in wireless mesh networks. *Comput. Secur.* (2013).

[15] Lucas Maciel, Alice Oliveira, Riei Rodrigues, Williams Santiago, Andresa Silva, Gustavo Carvalho, and Breno Miranda. 2022. A Systematic Mapping Study on Robotic Testing of Mobile Devices. In *2022 48th Euromicro Conference Series on Software Engineering and Advanced Applications (SEAA)*. IEEE, 475–482. https://doi.org/10.1109/SEAA56994.2022.00079

[16] Ke Mao et al. 2017. Robotic Testing of Mobile Apps for Truly Black-Box Automation. *IEEE Software* (2017).

[17] Gerard Meszaros. 2007. *xUnit Test Patterns: Refactoring Test Code.* Addison-Wesley.

[18] Larry Peterson, Thomas Anderson, David Culler, and Timothy Roscoe. 2002. A Blueprint for Introducing Disruptive Change in the Internet. In *Proc. ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)*. ACM, Cambridge, MA, USA, 1–6.

[19] Ju Qian et al. 2020. RoScript: A Visual Script Driven Truly Non-Intrusive Robotic Testing System. In *Proceedings of the 42nd International Conference on Software Engineering*.

[20] Chenhao Wei, Lu Xiao, Tingting Yu, Sunny Wong, and Abigail Clune. 2024. How Do Developers Structure Unit Test Cases? An Empirical Analysis of the AAA Pattern in Open Source Projects. *IEEE Transactions on Software Engineering* 51 (2024), 1007–1038. https://doi.org/10.1109/TSE.2025.3537337

[21] Xue Zhang et al. 2015. Ticket-Based Authentication for Fast Handover in Wireless Mesh Networks. *Wireless Personal Communications* (2015).

[22] Han Zou, Chun-Lin Chen, Maoxun Li, Jianfei Yang, Yuxun Zhou, Lihua Xie, and Costas J. Spanos. 2020. Adversarial Learning-Enabled Automatic WiFi Indoor Radio Map Construction and Adaptation With Mobile Robot. *IEEE Internet of Things Journal* 7, 8 (2020), 6946–6954. https://doi.org/10.1109/JIOT.2020.2979413