Drone-Based Communication Networks Using Raspberry Pis and ESP32s

By: Sahith Bodla and Joshua Neela

Abstract

This paper explores developing and evaluating a drone-to-drone communication network using Raspberry Pis and ESP32 microcontrollers using ESP-NOW in a component-connected system representing network drones. The primary goal is to test these devices' reliability, scalability, and performance in wireless networking applications. This work addresses the challenge of maintaining robust communication in dynamic and potentially obstructed indoor and outdoor environments, such as large-scale search-and-rescue missions or delivery systems. By leveraging multi-hop ad hoc networks and hybrid communication strategies, we evaluate the performance of Wi-Fi and MAC address-based communication for signal propagation across varying distances and obstacles. Our results demonstrate the feasibility of using low-cost hardware for establishing scalable and efficient communication networks in drone swarms, while also highlighting areas for improvement in throughput and latency.

Introduction

Drones have revolutionized fields such as disaster management, delivery logistics, and environmental monitoring. However, effective coordination among drones requires robust communication systems capable of adapting to dynamic environments and maintaining low latency. Existing solutions rely heavily on costly hardware or centralized infrastructure, which are often impractical in remote or resource-constrained areas.

We propose a low-cost, decentralized communication system using Raspberry Pis and ESP32 microcontrollers to enable reliable drone-to-drone communication. Our approach focuses on leveraging multihop ad hoc networks and hybrid communication models to address the limitations of existing centralized systems. By integrating Raspberry Pis for high-bandwidth Wi-Fi communication and ESP32s for lightweight MAC-based direct communication, we create a flexible system that adapts to varying operational demands.

This problem is critical as drones are increasingly deployed in scenarios requiring decentralized and adaptable communication. Existing research has explored wireless communication for drones, but few focus on achieving high reliability with low-cost hardware. Our contribution demonstrates how commodity devices can achieve robust communication for drone swarms, offering a scalable solution for real-world applications. Additionally, apart from contributions from researchers like Luca Mattola, there is a limited amount of existing research that dives into exploring the capabilities of drone communication within indoor environments.

Our goal with this project is to determine which method of ad-hoc drone communication is most effective in ensuring low latency for deploying an ad-hoc drone network. Ensuring a robust drone wifi network can ensure effective network response across multiple different use cases for a drone network in areas and situations where there is poor traditional networking infrastructure. One prominent example would be in emergency response, where a sturdy drone network can be leveraged to provide network connectivity across a disaster zone in the event of a natural disaster such as a hurricane. Other relevant applications of a drone network would be rural connectivity, environmental monitoring, and high-reliability communication. Overall, this begs the need to understand what modes of drone-to-drone communication are most effective at ensuring strong inter-drone communication despite various sources of interference that occur from obstacles and signal interference in an urban environment.

Thus, we are trying to determine which methods of wireless communication are most efficient and reliable to establish drone-to-drone communication under various conditions and altitude differences. Additionally, we also want to determine to what extent the signal strength of the drone communication begins to fade as we continue to increase the distance between both drones.

Contributions

Sahith was responsible for the ESP32 code and setup, ensuring reliable communication using the ESP32 hardware. Josh focused on the Raspberry Pi code and setup, enabling high-bandwidth communication and data collection functionalities. Both Sahith and Josh worked together on data collection, ensuring comprehensive testing across different environments.

For the written components, Josh contributed to the Introduction, Motivation, Further Directions, and Conclusions, as well as the creation of relevant data graphs. Sahith took the lead on the Abstract, Approach, Data Evaluations, and References, while also collaborating with Josh on generating data graphs.

Motivation and Background

Before exploring different challenges, it is worth noting that there are multiple metrics and parameters that we would need to keep in mind. The first is Received Signal Strength (RSSI). This represents the strength of the signal we initially receive at each node. The congestion window is also something that should be considered as it represents the TCP flow control mechanism, determining how much data can be in flight before an acknowledgment is received. This directly impacts the network's performance under varying conditions. Both of these metrics are good indicators of overall data reliability in drone-to-drone communication. Two other metrics that are important to keep note of are throughput and latency. Throughput in this instance would be the amount of data successfully transmitted in Kbps or Mbps. Latency represents the delay in overall data packet transmission. Ideally, higher throughput and low latency are incredibly important considering the variable applications of drone networks. Some notable parameters that need to be considered when evaluating each of these metrics would be the altitude and lateral distance of the receiving drone from the base drone.

The first major challenge is the environment as the particular environment in which the drone network is released can significantly impact its overall performance and reliability. In indoor environments, Non-Line-of-Sight (NLoS) conditions caused by walls, ceilings, and other physical obstructions can lead to signal reflections and multipath interference. These factors reduce the SNR by increasing packet loss, latency, and throughput degradation. Additionally, materials such as concrete, metal, and glass absorb or reflect Wi-Fi signals differently, creating unpredictable propagation patterns. In outdoor environments, factors such as wind, temperature variations, and moisture introduce instability. Wind obstructions can cause drones to move unpredictably, resulting in signal misalignment or inconsistent connectivity [4]. In addition, temperature changes can affect the efficiency of both battery performance and electronic components, while longer communication distances outdoors amplify free-space path loss, further attenuating signal strength.

The second major challenge for drone-based networks is the dynamic topology of the drones, meaning the positions of drones are constantly changing during flight. Unlike static networks, where nodes remain fixed, drones move relative to one another, requiring frequent updates to routing tables and network configurations to maintain connectivity. For example, as drones adjust their altitude or move across a field, their connections to other nodes may strengthen or weaken. This would require a real-time link adaptation. This dynamic behavior creates challenges in maintaining stable communication links and ensuring efficient data transmission, especially when drones move at varying speeds or under environmental disturbances [3].

The last major challenge facing drone-based networks is scalability. For drone-based networks, scalability is a crucial issue as the system must be able to accommodate more drones without seeing a noticeable drop in performance. Due to overlapping frequency bands, interference between Wi-Fi signals intensifies as the number of drones rises, resulting in lower throughput and higher delay [1]. Efficient resource allocation is also necessary for managing numerous nodes in order to guarantee that processing power, energy, and bandwidth are dispersed uniformly throughout the network. Furthermore, scaling makes the problem of dynamic topology worse because every drone adds new linkages and routes that need to be taken into consideration for the network to function.

Regarding related work, there are multiple other proposed solutions to improve drone network communication including centralized systems and mesh network protocols. Centralized systems rely on fixed ground stations or cellular infrastructure. In urban and suburban settings with pre-existing infrastructure, centralized systems offer dependable performance. They make it possible to centrally control drone networks, guaranteeing strong data processing capabilities, steady connectivity, and high-bandwidth transmission. For instance, centralized control can optimize drone placement for network coverage [5]. These systems are not appropriate for rural locations or catastrophe regions where communication towers are either nonexistent or destroyed, though, because they are intrinsically limited by the available infrastructure. Furthermore, centralized methods have a single point of failure, which means that the entire communication network may stop working if the central node becomes unavailable. These restrictions impede the flexibility and robustness required for drone-based Wi-Fi networks in our setting, especially in dynamic or uncertain settings like expansive open fields or inside testing areas. Our method fills the gap where conventional centralized systems fail by offering a hybrid model that combines direct MAC-based communication for lightweight, energy-efficient communication with centralized coordination for high-bandwidth workloads.

Additionally, while centralized systems provide reliable coordination and high-bandwidth communication, their reliance on fixed infrastructure makes them unsuitable for dynamic or remote deployments. As an alternative, mesh networks are often implemented in drone-based systems to enable autonomous, multi-hop communication between drones. However, mesh networks encounter significant challenges when applied to drone swarms, particularly in large outdoor environments where maintaining connectivity is critical. Another major limitation is hardware and cost. Mesh networks require specialized communication hardware and protocols to support multi-hop transmissions, significantly increasing deployment costs. This makes them impractical for large-scale or cost-sensitive drone operations [3]. Energy efficiency is another critical concern. In mesh topologies, drones must continuously relay data between nodes, acting as both transmitters and routers. This dual role introduces substantial processing and energy overhead which exacerbates battery limitations already imposed by flight duration and environmental conditions such as wind resistance and altitude changes. Overall, drone energy constraints are a significant bottleneck when considering sustained multi-hop communication [3]. This aligns with similar concerns mentioned in the DroneBridge Github repo, where excessive computational overhead can impact drone reliability and lifespan [2].

Furthermore, routing challenges in mesh networks are amplified by drone mobility and dynamic link quality. The constantly shifting topology of drone swarms makes it difficult to maintain stable multi-hop connections, particularly in large outdoor fields where nodes are spread over wide distances. Network performance is significantly impacted by packet losses, reduced throughput, and increased latency caused by high drone density, environmental interference, and connection failures [3]. These scalability issues make mesh networks less suitable for drone swarm applications that demand low-latency, high-reliability communication.

While mesh networks provide theoretical advantages for ad hoc networks, their practical implementations introduce energy inefficiencies, high costs, and connectivity challenges. These drawbacks highlight the necessity of our hybrid strategy, which blends lightweight MAC-based protocols for energy-efficient, short-range communication with the centralized coordination needed for high-bandwidth transactions. Even in dynamic and uncertain environments, this combination guarantees cost-effectiveness, scalability, and network resilience.

Our hybrid approach effectively combines the strengths of centralized systems, mesh networks, and lightweight communication protocols to address their respective limitations in drone-based wireless networks. Centralized systems excel at providing high-bandwidth connectivity and robust data aggregation but are constrained by infrastructure requirements and susceptibility to single points of failure. On the other hand, mesh networks offer flexibility in dynamic environments but struggle with scalability, energy overhead, and routing complications, particularly in outdoor deployments [3].

Our solution bridges these gaps by leveraging centralized systems for tasks requiring high data throughput, such as monitoring network performance metrics like bit rate, latency, and throughput. Simultaneously, it integrates underutilized ESP32 ESP-NOW MAC-layer communication in addition to Raspberry Pi Wi-Fi communication to enable direct, energy-efficient drone-to-drone links for lightweight tasks. This dual-layered approach ensures that lightweight protocols manage energy-constrained, short-range communication while centralized coordination benefits high-bandwidth workloads.

By combining these elements, our approach achieves a cost-effective, scalable, and energy-efficient network capable of adapting to both outdoor and indoor deployments. The use of Wi-Fi for high-performance tasks, in addition to ESP32-based communication for low-power operations, ensures that our network operates reliably under dynamic conditions without incurring the high costs and energy inefficiencies associated with mesh-only or centralized-only systems. This unique balance makes our solution particularly well-suited for drone swarms deployed in large, remote fields or hazardous environments where infrastructure is limited and energy optimization is crucial.

Approach

Due to the lack of drone strength to fly a system of components over half a pound, we decided to represent each “drone” in the ad hoc network as the system of wireless communication components without the actual drone, instead varying the altitudes and distances of the receiver system position to mimic the drone’s movements and positioning.

The system comprises a hybrid communication architecture utilizing Raspberry Pis and ESP32 microcontrollers to establish the ad hoc network. The Raspberry Pis are configured as Wi-Fi nodes operating over the IEEE 802.11 standard to ensure high-bandwidth communication, particularly suited for tasks demanding larger data transfers. Complementing this, ESP32 microcontrollers facilitate MAC-based peer-to-peer communication using ESP-NOW. These two components are powered by a standard portable battery for consumer phones. Together, these components form the backbone of a multihop ad hoc network, where each system of components “drone” functions as a dynamic router to forward packets between each other, enabling seamless and decentralized communication. Additionally, a personal hotspot is created to act as a base station for each of the system nodes to connect with each other.

In our testing, we decided to measure various different metrics to gauge overall network strength. Between the two-node system, we measured the connection’s bit rate, congestion window, number of retries, number of pings, latency, and throughput. We also measured RSSI between the client system and the base station. For our testing environment, we decided to test the network connectivity of the component systems across three different environments. For the first environment, we measured the impact of an indoor environment. We tested inside the 410 E. Green 3x2 layout, adjusting the distance between each component system from 0-40 ft. For outdoor testing, we evaluated the difference for our metrics floor by floor. Starting from the ground floor to floor 5 with 9 ft increments, measurements were taken for 1 minute each for each of the elevations. Additionally, an additional measurement was taken 40 ft away in the opposite direction of the initial source location in addition to a 45 ft altitude height. This can be represented as an additional 61 ft measurement for our outdoor data. There was also additional obstruction in additional obstacles such as railings and flooring. For our final environment, we recorded measurements with varying altitudes in the stairwell to simulate signal reflections and an increased amount of obstructions present in urban environments. The altitudes are the same heights as the outdoor environment recordings.

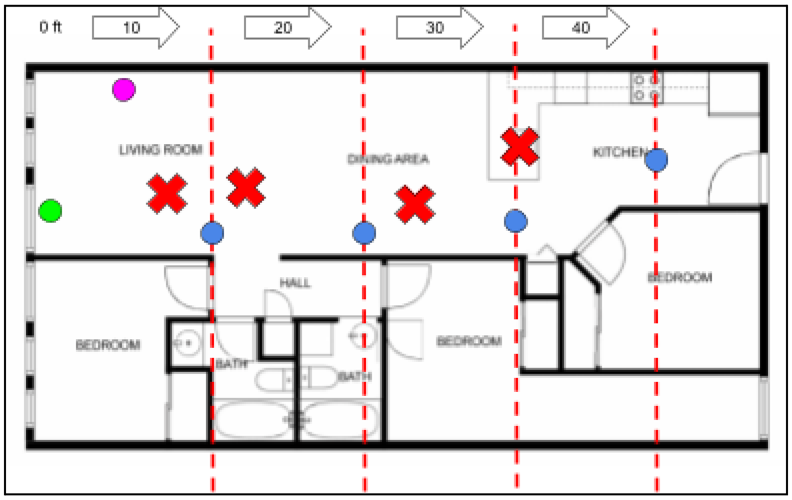

Figure 1: Indoor environment:

The green dot represents the server drone and its respective Piand ESP32 sitting stationary in the corner of the living room. The blue dot represents the client drone and its respective Piand ESP32 that will be positioned at distances starting from 0 feet (right next to the server) to 40 feet. The 10-ft increments will be done at approximately 1-minute intervals. The red X's represent obstructions in the line of sight between the client and the server, which we hypothesize may affect the performance metrics of the communication. The pink dot represents a secondary base station (laptop) connected to the phone providing the hotspot WiFi connection that the client pings as well.

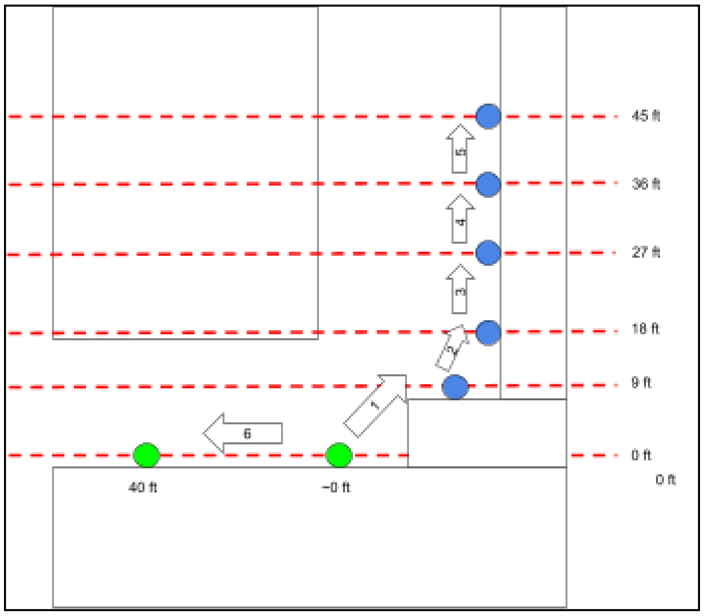

Figure 2: Outdoor environment where each altitude difference is approximately 9ft.
Same legend as the previous figure. First, the client drone will increase its altitude across 5 floors going from 0 feet (right next to the server) to 45 feet. Lastly, the server will move backward further away from the client hovering at the 5th floor to see the nature of communication over the maximum distance away from the client.

<u>Experimental Results</u>

Data Overview:

In each group of data, the first six graphs utilize iperf3 to illustrate the performance metrics of the network. Specifically, the first four graphs provide detailed insights into client-to-server performance, capturing key parameters such as throughput (data transfer rate), bitrate, the number of retries observed in the TCP protocol, and the size of the TCP congestion window. These metrics collectively provide a comprehensive view of the network's ability to maintain stable performance across varying distances in the indoor environment. They also focus on the discrepancies in transfer rates and bit rates between sender-to-receiver and receiver-to-sender directions.

The next two graphs in each section focus on ping latencies and ESP32 RTT when sending over a small “hello_world.txt” and a large 10 MB file. The first 6 graphs for each environment represent data collected from the Raspberry Pis and the last graph represents data collected from the ESP32s with ESP-NOW.

Indoor data:

Figure 3

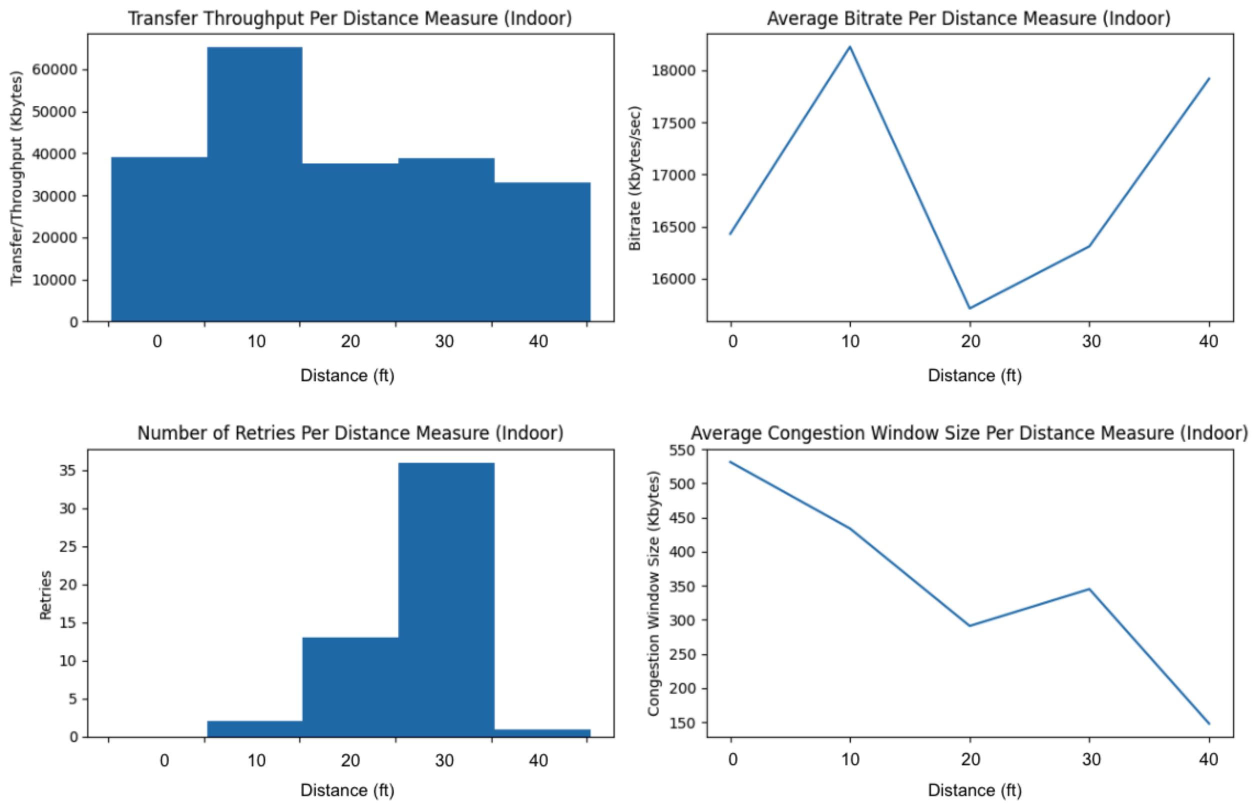

Figure 4

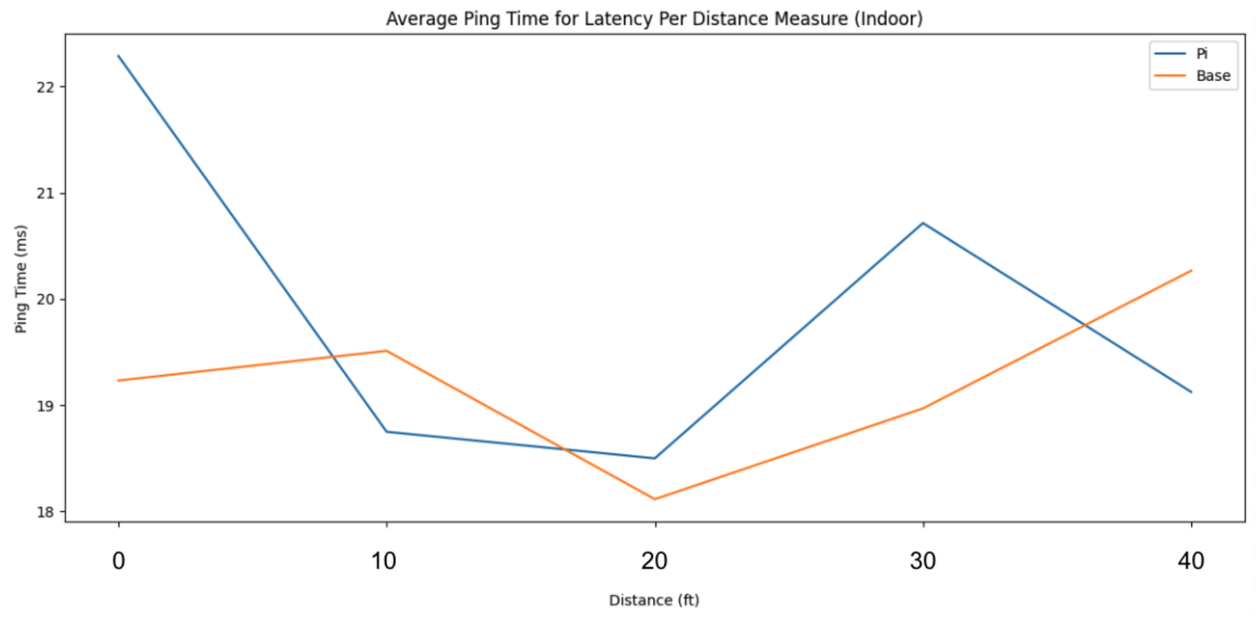

Figure 5

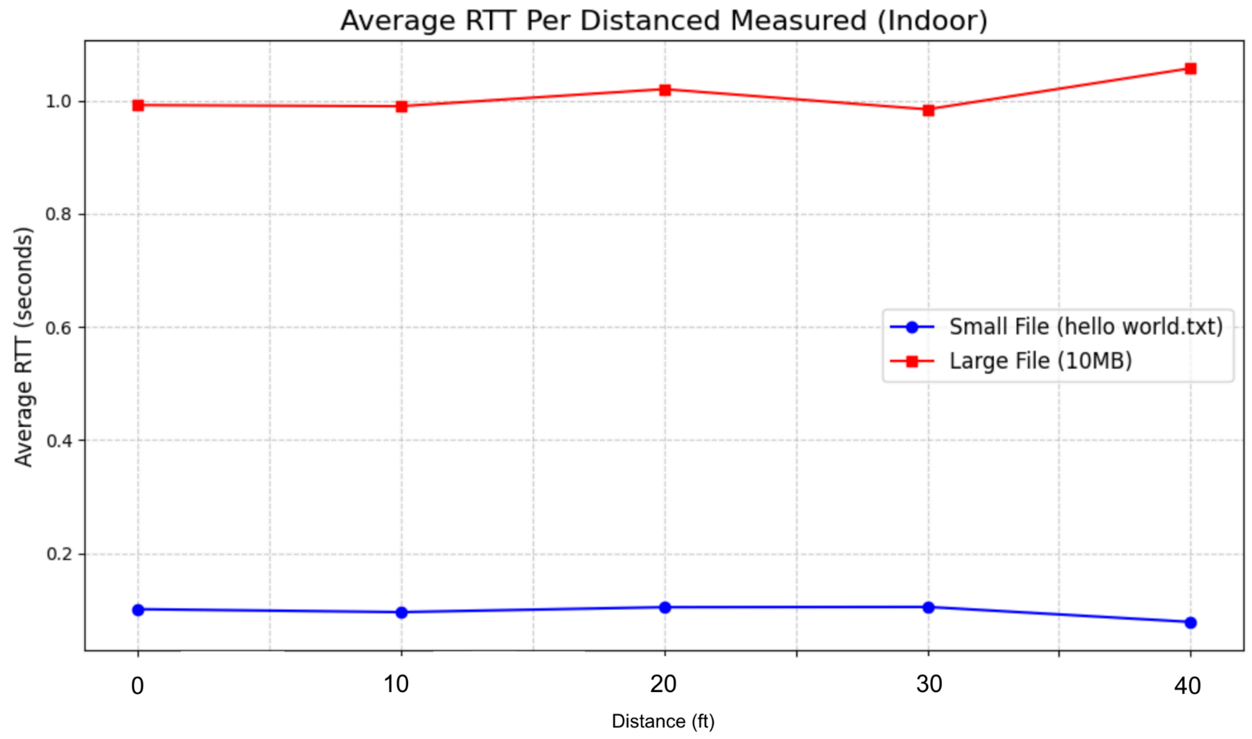

Figure 3: These four graphs highlight the total sum of the throughput and the number of retries in addition to the average bitrate and congestion window for the communication across 10-foot increments in distances when transferring 250-byte chunks of a 1MB file for our indoor trials. Looking at the graphs there is a correlation between the increase in distance and the number of retries as well as a decrease in average bitrate as distance increases. For the remaining graphs, there does not seem to be any strong pattern with the increase in distance and the sum of the remaining metrics. This can be attributed to the

fact that there were few obstructions in the indoor space with only a maximum of 40 feet in between the two nodes.

Figure 4: This graph displays the average ping time per distance measure. The pings are used to measure the round-trip time for 64-byte packets to travel back and forth outside. Thus, higher ping times reflect greater latency. There appears to be no discernible correlation between distance and ping time as the environment is relatively short and obstructions between them likely caused the seen variations and fluctuations.

Figure 5: This compares the average RTT per distance measured between the large file of 10MB and the small file size of 13 bytes for “hello world!” for the indoor trials. The large file is sent in 250-byte chunks. There are generally consistent average RTTs for both the small file and the large file. This is likely due to the abundance of Line of Sight communication.

Outdoor data:

Figure 6

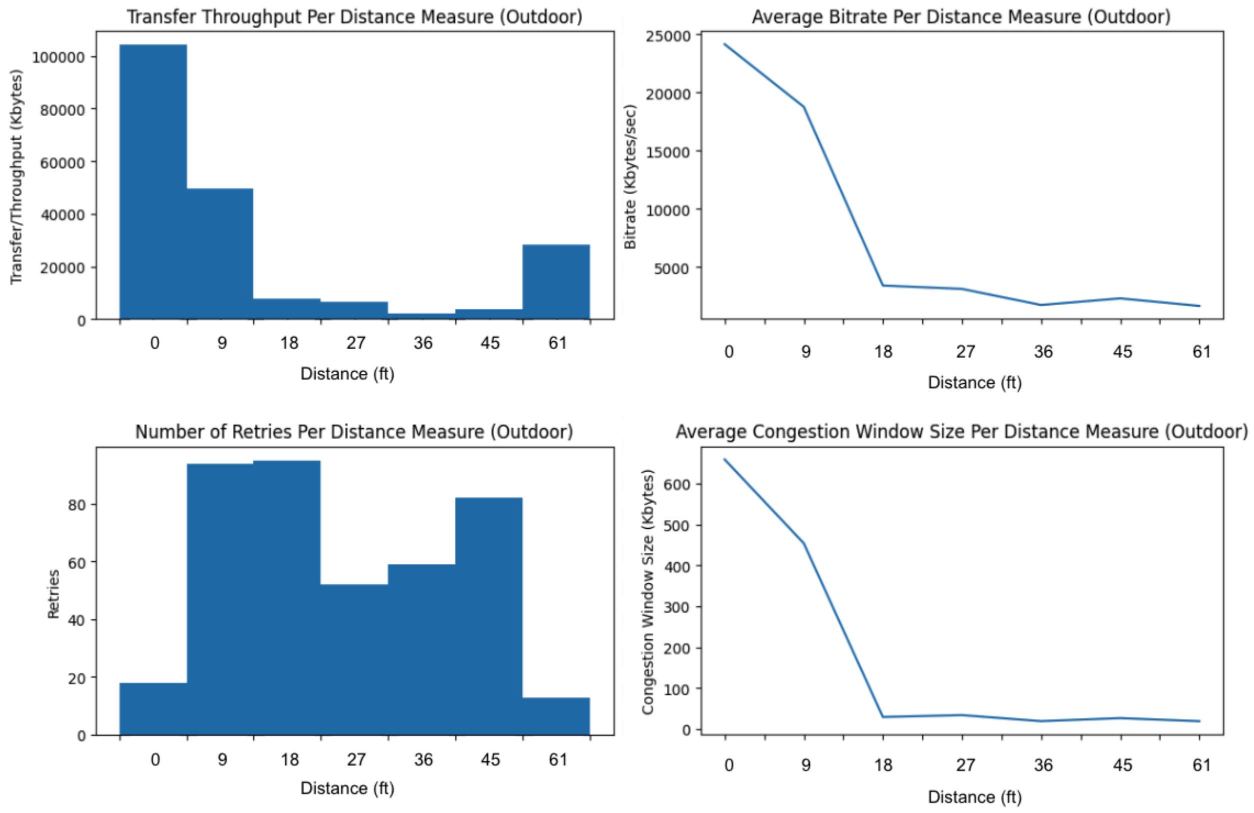

Figure 7

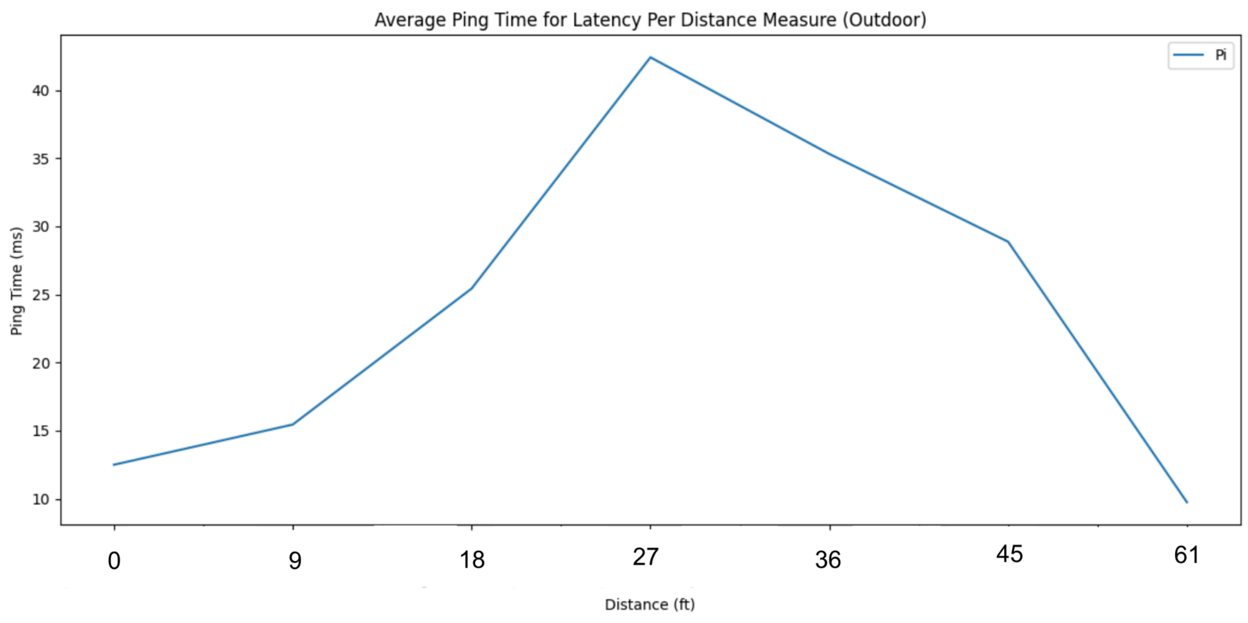

Figure 8

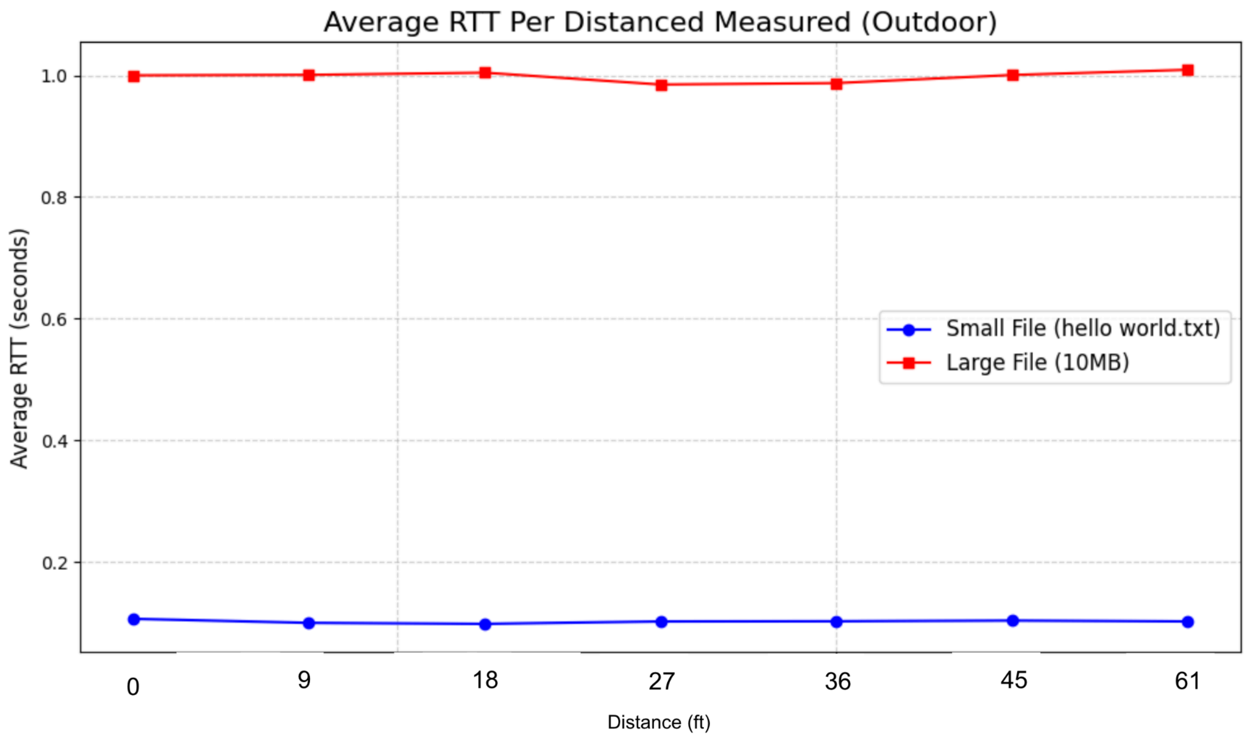

Figure 6: These four graphs highlight the total sum of the throughput and the number of retries in addition to the average bitrate and congestion window for the communication across 9-foot increments in altitudes with an additional 61 ft metric when transferring 250-byte chunks of a 10MB file for our outdoor trials. It is noticeable that throughput, average bitrate, and average congestion window decrease over time as distance is increased. This is a sign that signal degradation occurs as altitude increases.

Figure 7: This graph displays the average ping time per distance measure. The pings are used to measure the round-trip time for 64-byte packets to travel back and forth outside. Thus, higher ping times

reflect greater latency. There appears to be an initial correlation between an increase in distance and an increase in average ping time up to 27 feet. However, the opposite correlation occurs after this measurement which leaves possible conclusions from this graph to be inconclusive.

Figure 8: This compares the average RTT per distance measured between the large file of 10MB and the small file size of 13 bytes for “hello world!” for the outdoor trials. The large file is sent in 250-byte chunks. There are consistent average RTTs for both the small file and the large file. This is likely due to the abundance of Line of Sight communication.

Stairwell data:

Figure 9:

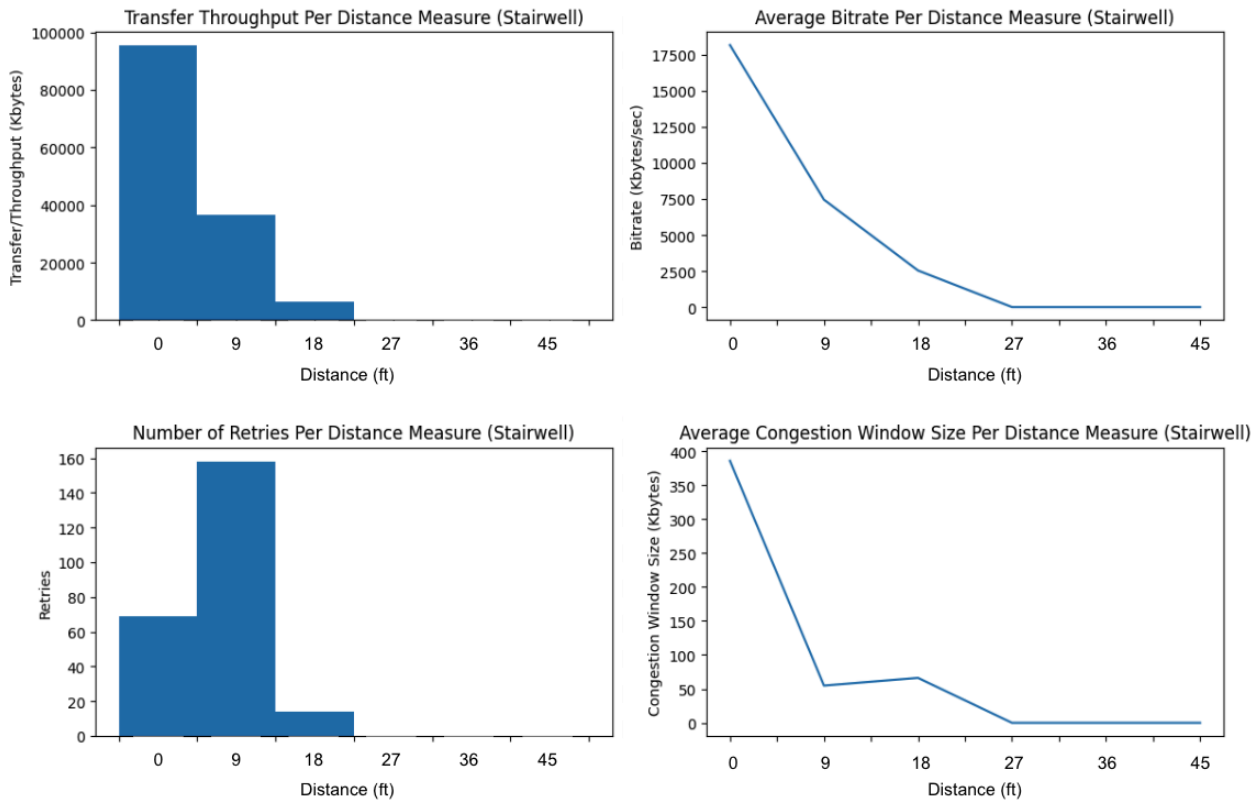

Figure 10:

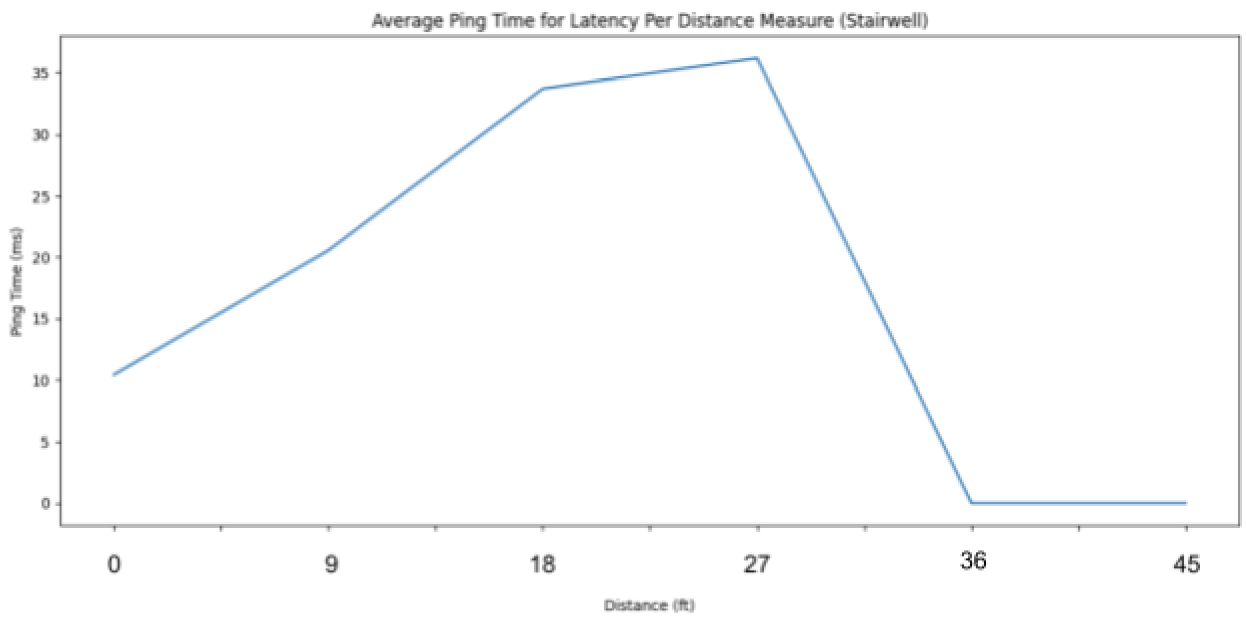

Figure 11:

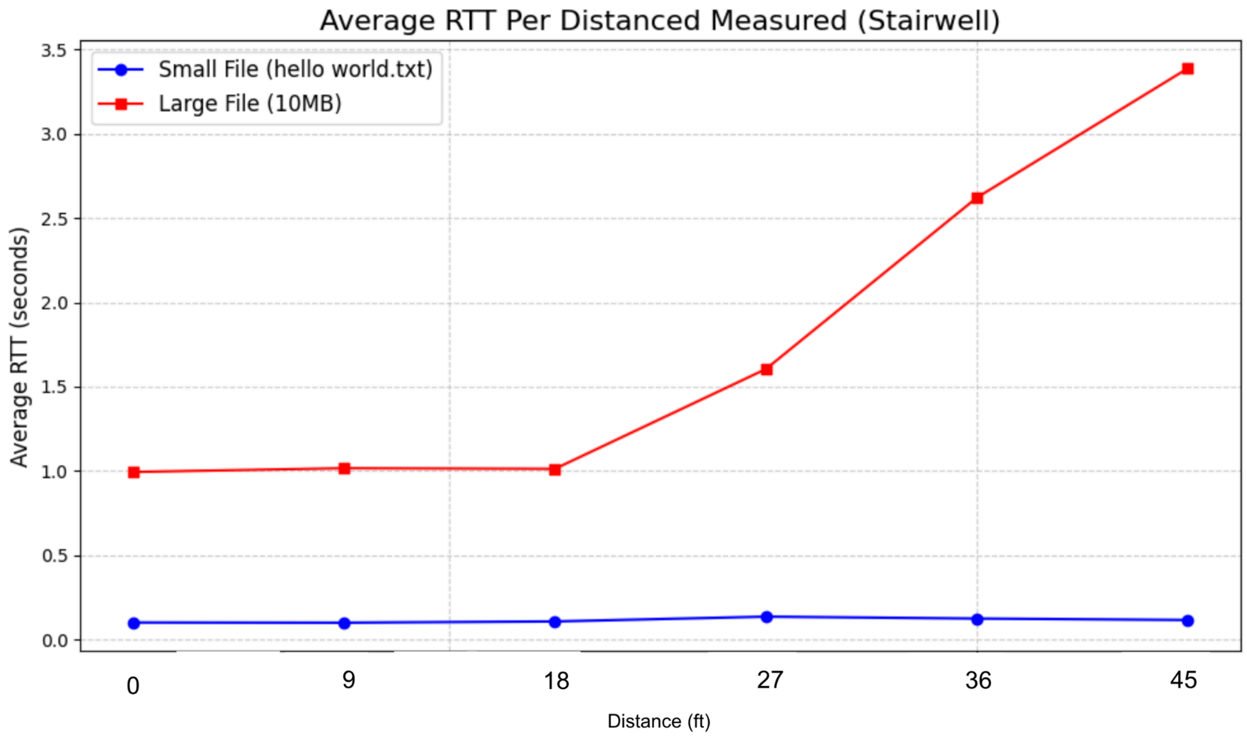

Figure 9:
These four graphs highlight the total sum of the throughput and the number of retries in addition to the average bitrate and congestion window for the communication across 9-foot increments in altitudes when transferring 250-byte chunks of a 10MB file for our stairwell trials. It is worth noting that the receiver node lost connection after the 18th-foot altitude distance. This is likely due to a high amount of obstructions present within the stairwell such as stairs, walls, floors, and railings. Apart from that, there is a clear correlation between a decrease in throughput and a decrease in average bitrate as the distance between the nodes increases.

Figure 10:
This graph displays the average ping time per distance measure. The pings are used to measure the round-trip time for 64-byte packets to travel back and forth. Thus, higher ping times reflect greater latency. There is a correlation between an increase in distance and an increase in average ping time. The average ping time drops to 0 at 36 feet altitude when there is a loss in communication between the source node Raspberry Pi and the receiver node Raspberry Pi.

Figure 11: This compares the average RTT per distance measured between the large file of 10MB and the small file size of 13 bytes for “hello world!” for the outdoor trials. The large file is sent in 250-byte chunks. While the small file has a consistent average RTT, there is an increase in the average RTT as altitude increases past 18 feet in the stairwell. The increase in average RTT is likely due to the excess increase in obstructions and interference past a certain point of 18 feet altitude in the stairwell.

RSSI over distances:

Figure 12:

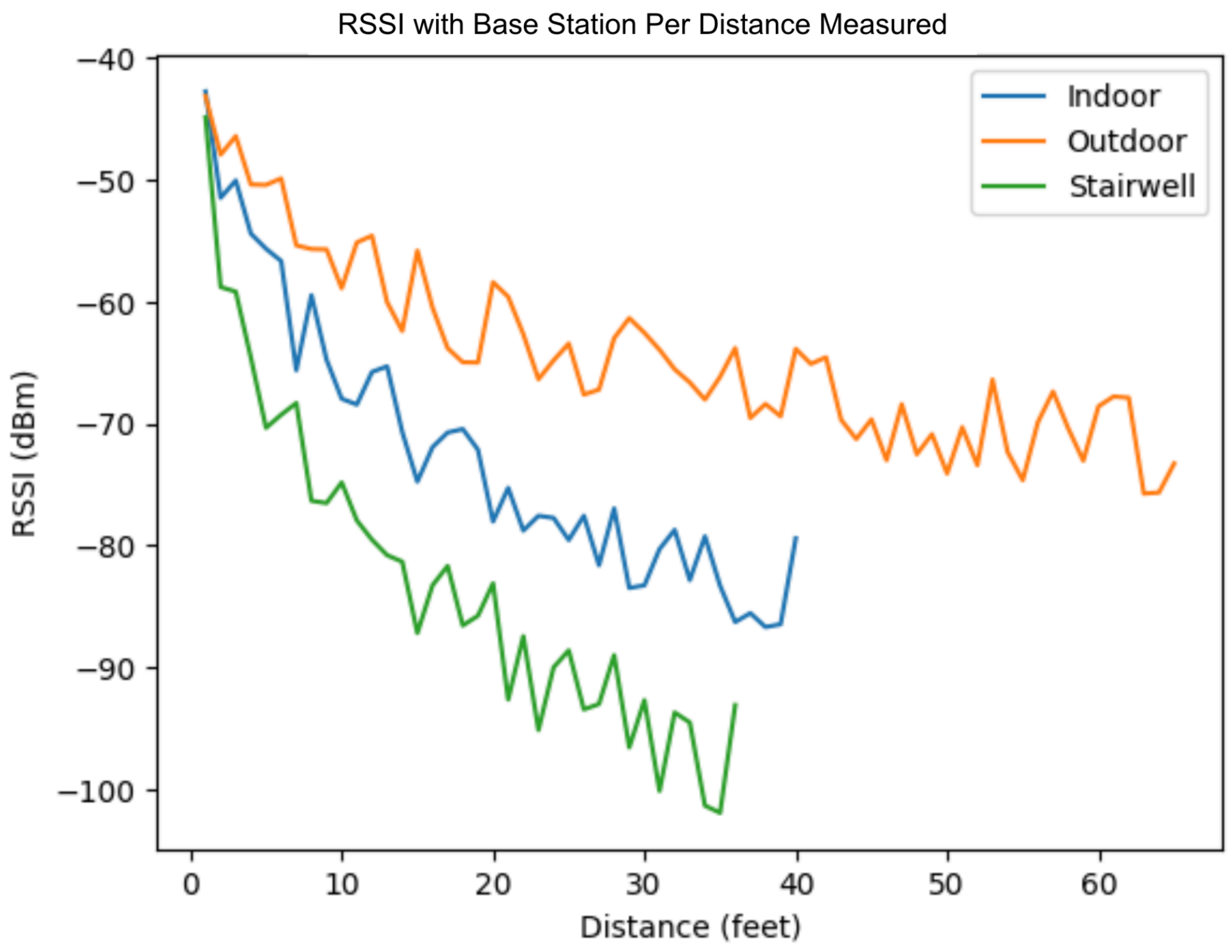

Figure 12: The RSSI trends are inherently declining over larger distances between the drones. The largest drop-off occurs in the stairwell, as the connection is completely lost after the drone ascends to the fourth floor. The outdoor environment has the least amount of obstacles and obstructions, preserving an unperturbed line of sight, so the RSSI drop-off isn’t very drastic.

<u>Conclusions and Future Directions</u>

Through this project, we demonstrated the feasibility of creating robust, cost-effective drone-to-drone communication networks using low-cost hardware like Raspberry Pis and ESP32 microcontrollers. By incorporating a hybrid communication model that combines Wi-Fi via Raspberry Pis for high-bandwidth tasks and MAC-based ESP32 links for long-range communication, the system adapts well to varying operational demands.

Our experiments revealed critical insights into network performance across diverse environments, including indoor, outdoor, and obstructed stairwell scenarios. Notably, we observed that ESP32 connections were significantly more persistent in dense environments where the Raspberry Pis lost connectivity. This highlights the ESP32's reliability for maintaining communication links in challenging conditions, such as how in the stairwell the connection persisted despite increasing RTT, making it well-suited for long-range, stable communication tasks. Conversely, the Raspberry Pis provided a more versatile platform, allowing us to collect much more diverse data and metrics regarding the connection status, such as throughput, latency, and retries. This ability makes the Pis valuable for network diagnostics and performance analysis.

While Wi-Fi communication excels at providing higher throughput and more comprehensive data, it suffers significantly in dense and obstructed environments due to signal attenuation and interference. In contrast, the ESP32-based communication proved to be more resilient in such conditions, ensuring sustained connectivity even when Wi-Fi links failed.

From this, we learned that Raspberry Pis can be effectively used for data collection and network analysis, whereas ESP32s serve as a reliable backbone for maintaining persistent, long-range communication links. This hybrid approach lays a solid foundation for adaptive and resilient drone-based networks suited for dynamic, real-world applications such as the applications discussed earlier in disaster recovery, rural connectivity, and environmental monitoring

Future research can focus on scaling the network by adding more drones to evaluate various network structures such as ring-based topologies, quorum-based communication, and energy-efficient configurations. It is also worth exploring if these results are reproducible with the wireless communication system mounted on top of stronger, more advanced drones. Exploring different communication protocols tailored for these structures can optimize performance and energy consumption. Additionally, investigating other communication mediums like cellular networks or LoRa can provide alternatives for long-range, low-power scenarios. We may also attempt looking into different hardware that is also lightweight e.g. Adafruit Feather. Research can also enhance network reliability by integrating multi-frequency and adaptive communication systems. Utilizing dual-band Wi-Fi (2.4 GHz and 5 GHz) with intelligent algorithms can help mitigate signal attenuation and interference issues. Additionally, implementing dynamic frequency management techniques will further optimize connectivity, especially in environments with high interference or non-line-of-sight conditions.

References:

[1] M. Bohmer, A. Schmidt, P. G. Pereira, and Thorsten Herfet, “Latency-aware and -predictable Communication with Open Protocol Stacks for Remote Drone Control,” *arXiv* (Cornell University), pp. 304–311, May 2020, doi: https://doi.org/10.1109/dcoss49796.2020.00055.

[2] DroneBridge, “DroneBridge/README.md at master · DroneBridge/DroneBridge,” *GitHub*, 2017. https://github.com/DroneBridge/DroneBridge/blob/master/README.md

[3] A. Guillen-Perez, R. Sanchez-Iborra, M.-D. Cano, J. C. Sanchez-Aarnoutse, and J. Garcia-Haro, “WiFi networks on drones,” *2016 ITU Kaleidoscope: ICTs for a Sustainable World (ITU WT)*, Nov. 2016, doi: https://doi.org/10.1109/itu-wt.2016.7805730.

[4] A. I. Hentati and L. C. Fourati, “Comprehensive survey of UAVs communication networks,” *Computer Standards & Interfaces*, vol. 72, p. 103451, Oct. 2020, doi: https://doi.org/10.1016/j.csi.2020.103451.

[5] N. I. Sarkar and S. Gul, "Artificial Intelligence-Based Autonomous UAV Networks: A Survey," *Drones*, vol. 7, no. 5, p. 322, May 2023, doi: https://doi.org/10.3390/drones7050322.

Code: https://github.com/bodlasahith/aerial-drone-wifi-network