A Unified Real-Time Benchmark To Evaluate the Practical Impact Of Wireless Technology on the Edge

A Thesis Presented for the
Master of Science
Degree
The University of Tennessee, Knoxville

Benjamin A. Clark

December 2025

© by Benjamin Clark, 2025
All rights reserved.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisors, Dr. Aly Fathy and Dr. Mongi Abidi, for their invaluable guidance, support, and encouragement throughout my Master of Science in Computer Engineering at the University of Tennessee. Their insight, mentorship, and patience have been instrumental in shaping both this research and my growth as an engineer.

I would also like to acknowledge the Center for Hardware and Embedded System Security and Trust (CHECCS) IUCRC and our industry partner Phelps 2020, with special thanks to Dr. Mongi Abidi for facilitating this collaboration and providing the funding that made this work possible.

Finally, I am grateful to the AT&T 5G MEC Lab for offering access to state-of-the-art infrastructure and a unique opportunity to gain hands-on experience with real-world 5G systems. This exposure greatly enhanced the practical relevance and depth of this research.

To all those who have supported me along this journey—faculty, colleagues, friends, and family—thank you for your encouragement and belief in my efforts.

ABSTRACT

This thesis presents the design, implementation, and evaluation of a portable, real-time edge computing and streaming system deployed over modern wireless network technologies, including Wi-Fi 6, 4G LTE, and 5G. Built upon a high-demand critical service use case, the system benchmarks the practical performance limits of compact, off-the-shelf hardware in realistic operating conditions. By varying processing platforms, communication channels, and application parameters, this study monitors both application-level performance and the state of the underlying network in real time.

The primary contribution of this work is the development of a benchmarking framework that quantifies how well consumer-grade edge devices meet the stringent requirements of critical infrastructure applications such as those in automation, manufacturing, and security. The system captures live sensor data, applies task-specific processing, streams the resulting media across a network link, and logs performance metrics including bitrate, latency, and frame rate. Simultaneously, it observes key link-layer indicators such as signal strength and noise levels.

A secondary but significant contribution is the characterization of how network conditions directly influence the performance of delay- and bandwidth-sensitive applications. For example, experiments with a 5G mmWave channel revealed that even minor increases in signal attenuation can cause substantial degradation in streaming bitrate and delay. These findings provide actionable insight into the design constraints and expectations for wireless deployment of critical services.

Additional contributions include improvements to system performance through the refinement of streaming protocols and compression techniques. These optimizations further enhance the system's ability to reflect real-world usage scenarios and make the findings more generalizable.

In summary, this thesis offers a practical and reproducible approach to benchmarking edge computing systems under real-world conditions, providing valuable guidance for engineers and researchers developing next-generation networked applications for critical infrastructure.

TABLE OF CONTENTS

Chapter One Introduction 1

Motivation 1

Pipeline of Thesiss 3

Thesis Contributions 4

Contribution 1 (Major) 4

Contribution 2 (Major) 5

Contribution 3 (Major) 5

Contribution 4 (Minor) 5

Synopsis 6

Conclusion 7

Chapter Two Literature Review 9

Foundations of Edge Computing in Critical Infrastructure 9

Definition and Architectural Models of Edge Computing 9

Role in Critical Applications 11

Multi-access Edge Computing 12

Modern Wireless Network Technologies for Real-Time Applications 16

Wi-Fi 6 16

4G LTE 18

5G NR 19

Performance Benchmarking Methods for Edge and Network Systems 21

Benchmarking Objectives and Metrics 21

Existing Edge and Network Benchmark Frameworks 23

Instrumentation and Measurement Techniques 27

Real-World Relevance 28

Streaming Protocols and Compression Techniques for Wireless Edge Systems 28
Real-Time Streaming Protocols 28
Compression vs. Computation Trade-offs 29
Protocol and Codec Optimizations for Wireless Communications 29
Conclusion 29
Chapter Three Benchmarking System Design and Implementation 31
Overview of Benchmarking Framework 31
High-level Architecture 31
Functional Breakdown of System Components 33
Data Flow and Processing Pipeline 34
Critical Service and Network Monitoring System and Application 38
Hardware (Specs, Interfaces, Rational) 40
Video Streaming 42
Application-Specific Processing 48
System Controller Graphical User Interface 49
Performance Monitoring and Data Logging 49
Network Interfaces 51
Wi-Fi 52
4G LTE 53
5G mmWave 53
Loopback 54
Cyber-physical Space and Realistic Operating Conditions 54
Conclusion 55
Chapter Four Experimental Methodology 57
Experimental Setup 57

Constants 57
Baseline 58
Wi-Fi 59
4G 60
5G 60
Performance Metrics 62
Conclusion 62
Chapter Five Results and Discussion 64
Performance Across Processing Platforms 64
Raw Data Logged by System Controller 64
Post Analysis Probability Distribution 65
Performance Across Wireless Technologies 71
Raw Data Logged by System Controller 72
Post Analysis Probability Distribution 77
Performance Under Impairment 86
Discussion of Findings and Practical Implications 89
This Work vs. Established Frameworks / Standards 90
Comparative Insights on Platform Performance 93
Network Behavior Under Load and Resilience to Impairment 93
Implications for Edge System Design and Future Applications 94
Conclusion 94
Chapter Six Conclusion 96
Summary 96
Contributions 98
Major Contribution 1: Unified Benchmarking Framework Design. 98

Major Contribution 2: Quantitative Characterization of Edge System Behavior. 98
Minor Contribution: Protocol and Software Optimization. 99
Future work 99
Integration with Advanced Network Techniques 99
Integration of Diverse Sensing Node Types. 100
Maximizing Bandwidth and Network Capacity Utilization. 100
Scaling to Dense, Distributed Sensing Networks. 100
Broader Multimedia Workloads and Video Source Diversity. 101
Evaluation of Alternative Streaming Protocols 101
Deployment in Outdoor and Complex Network Environments. 101
List of References 102
Vita 107

LIST OF TABLES

Table 3.1: Size Weight Power and Cost (SWaP-C) Characteristics of Computing Platforms. 40
Table 5.1: Comparison of Edge Benchmarking Frameworks. 90

LIST OF FIGURES

Figure 1.1: IoT-connected smart agriculture sensors enable the IoT. This example smart farm taken from [8] demonstrates the communications scalability requirement of modern applications. 2

Figure 1.2: Evolution of vehicular sensors, computing tasks and applications from traditional to connected and autonomous vehicles. These large volume real-time advanced vehicular applications, taken from [9], illustrate the sheer number of sensing functions integrated with vehicles. 2

Figure 1.3 Thesis Pipeline of data capture, edge processing, network transmission, impairment injection, multimedia streaming, and analysis. The thesis is divided into two phases, Analysis and Application. The Analysis Phase details the variable conditions of the edge computing system along with the methods for interpreting the resulting data. In this phase, the live data source, processing platform, wireless network technology, channel impairment, and layer 4 protocols are varied to determine their impact on system performance. The Application Phase consists of the cyber-physical system which streams data from client to server and benchmarks the data stream performance. It will also offer real-time visualization of the benchmarking metrics. 4

Figure 2.1: High-level architecture of an edge computing system. Taken from [18], this diagram depicts the typical architecture for edge computing. It begins with the cost-effective, special purpose edge devices in the perception layer. Then data travels upstream through the network layer to the edge compute resources where higher-capacity processing exists. If needed data can travel further into the internet to access cloud resources. 10

Figure 2.2: General MEC Framework. Taken from [23], this figure gives a general architecture for edge computing and highlights its connection to end devices and cloud infrastructure. The edge devices offer services and computing closer to the end users mitigating upstream traffic to cloud resources. Furthermore, domains like IoT devices and sources of big data can allocate edge resources to get a speedup in response times over conventional web services in a large data center. 13

Figure 2.3: Multi-access Edge System Reference Architecture. Taken from [24], this diagram outlines the functional components and their interaction points of the MEC design. 14

Figure 2.4: Industrial System Classes and their Performance Requirements. Taken from [27], this figure highlights the varying needs of critical applications. Safety systems, for example, rely heavily on low latency and high reliability, but they have relatively lower requirements for data rate and scalability. 17

Figure 2.5: 5G NR mmWave PHY Scheduling Intervals: Adapted from work by Fezeu et.al., this figure offers insight into how 5G NR approaches dividing its resources for multiple users on the PHY-layer [33]. Using a centralized scheduling system, where the gNodeB controls all data transmissions (via PDCCH and PUCCH signaling) and leverages scalable numerology to accommodate diverse service needs. The key to reducing latency is the flexible subcarrier spacing, which shortens the fundamental scheduling unit (slot) from 1 ms (like 4G) down to 0.125 ms for high-frequency millimeter-wave (mmWave) deployments. Also described in their work, are external factors that increase latency, including the asymmetric nature of UL/DL scheduling, the dynamic adjustment of data rates, re-transmissions due to poor channel conditions, and the power-saving techniques. 21

Figure 2.6: Edge Performance Benchmark Classification. Taken from [16], this diagram illustrates three primary dimensions used to evaluate edge computing configurations: system under test, techniques analyzed, and benchmark runtime. The system under test encompasses the physical and digital components of the platform, including processing, management, and communication resources. Techniques analyzed refer to the application-level and deployment-level optimizations that influence performance. Benchmark runtime captures the operational context, including execution environment and physical or cyber-physical conditions. Together, these dimensions establish a comprehensive framework for benchmarking edge resources and networks, integrating both system design and real-world deployment factors..... 22

Figure 2.7: OpenStack Environment Architecture. Sourced from Dollinger et.al., OpenStack is a modular cloud operating system designed to deploy and manage large pools of compute, storage, and networking resources using decoupled APIs [35]. The architecture is organized around a Controller Node, which intercepts user requests, orchestrates the workflow across various services, and typically hosts the centralized dashboard. Compute Nodes manage the actual instances, supporting deployments like VMs, containers, and bare metal. This design separates component orchestration from device management, allowing for flexible and scalable infrastructure control..... 25

Figure 2.8: Example IoT Edge Architecture. This diagram was taken from the work done by Das et.al. [17]. EdgeBench is a specialized benchmark suite designed to evaluate the performance of applications in MEC environments using three canonical workloads: audio-to-text, image recognition, and a scalar sensor emulator. Its architecture involves edge devices sending data messages to a cloud-based IoT Hub (like AWS or Azure), which then routes the payload to storage. Metrics like compute time, time-in-flight, and end-to-end latency are captured using timestamps added at the edge device, the IoT Hub, and the final cloud storage to systematically measure performance across the entire pipeline. The framework is also adapted for cloud-only serverless pipelines to provide a baseline comparison for the edge-based execution 26

Figure 2.9: OpenRTiST Experimental Setup. Taken from the original paper by George et.al. [36]. The OpenRTiST architecture is a client-server design centered on the Gabriel framework for mobile cognitive assistance. A mobile client (the input) captures a live video feed and offloads frames to the edge server via the network. The server runs a Deep Neural Network (DNN) for real-time style transfer, which processes the frames and returns the transformed video (the output) to the client for display. This real-time interaction benchmarks end-to-end latency by measuring the time from video capture to the display of the processed, stylized frames..... 27

Figure 3.1: High-level System Design of the Benchmarking Framework. The architecture of this system utilizes three main components to benchmark system performance in a network edge scenario. The sensing node leverages an image sensor, SBC, and wireless modem to capture and transmit sensitive data. Depending on the capabilities and conditions of the intermediate network, the system controller will observe the incoming data stream at a measurable rate and delay. It is here that the user will visualize the live performance of the system..... 32

Figure 3.2: Functional Diagram of the Benchmarking Framework. The three main components of the benchmarking framework, namely the sensing node, network gateway, and system controller collectively participate in the experiment. The sensing node is responsible for video capture, metrics and performance monitoring, and streaming data. The network gateway facilities communication between the sensing node and system controller. The system controller receives streamed video, collects further metrics, and serves as the hub for visualization and control. 34

Figure 3.3: Data Flow and Processing Pipeline of the Critical Service Application. Data of the Critical Service Application flows through five main stages: data capture, processing, transmission, store, and analysis and visualization. The orange segments represent the sensing node, blue represents the network, and green represents the system controller. 36

Figure 3.4: Data Flow and Processing Pipeline of the Network Monitor Tool. Data of the Network Monitor Tool flows through four main stages: metrics capture, transmission, store, and analysis and visualization. The orange segments represent the sensing node, blue represents the network, and green represents the system controller. 36

Figure 3.5: Combined Data Flow and Processing Pipeline of the Benchmark Framework. This highlights the interoperation between the critical service and the network monitor tool. Data captured from each are synchronized before transmitting over the network allowing the system controller to display both in real-time. 37

Figure 3.6: Critical Service and Network Monitor Processes. This figure illustrates the main functions of the client and server processes for both the critical service and network monitor applications. For each application, the client executes on the sensing node while the server side executes on the system controller. 39

Figure 3.7: Processing Platforms of the Sensing Nodes. The devices shown in this figure are the processing platforms integrated into the sensing nodes. These platforms perform video capture, data stream management, and application specific processing. The top image depicts the Jetson Orin NX edge AI computer. On the bottom left is the Intel NUC i7-1360p mini computer. The Radxa Rock 5B RK3855 single-board-computer is shown in the bottom right image. 41

Figure 3.8:Processing Platform of the System Controller. The device depicted in this image is used as the processing platform for the system controller. The Dell OptiPlex 5060 is a general-purpose desktop system equipped with an Intel Core i7-8700 and 16GB DDR4 RAM. The system controller will display information and interact with users through its display connected via HDMI 2.0. 41

Figure 3.9: GStreamer Video Processing Pipelines for Client and Server. This figure gives more detail into what the GStreamer pipelines consist of within the sensing node and system controller. The client pipeline captures, encodes, and packetizes video frames into an RTP/UDP stream for transmission. Conversely, the server pipeline receives these packets, depacketizes the

payloads, decodes the data back into video frames, and provides the reconstructed stream to the main application for display and analysis. 43

Figure 3.10: GStreamer Pipeline Element Pad Probes. This figure highlights an indispensable mechanism that allows a parent application to probe aspects of the pipeline elements, such as buffered frames in memory and readable writable properties. Most importantly, for the streaming application used in this thesis, this allows for direct access to video frames enabling bounding boxes to be overlayed. Additionally, it allows for the modification of network packets..... 44

Figure 3.11: Embedding and Extracting Timestamps from RTP Packets. Building on Figure 3.10, this diagram shows at which elements RTP packets are accessed to embed and extract timestamps. These timestamps provide a temporal marker representing the moment in time that packets are sent for transmission. This method allows for the receiving device to calculate end-to-end delay..... 45

Figure 3.12: RTP Packet Fixed Header Fields. Taken from [39], this figure details the standard RTP packet header, which is essential for structuring and organizing the transmission of video frames over a network. The header contains a crucial customizable field, the RTP header extension, located at bit 4, which allows for additional information to be included in the packet. 47

Figure 3.13: RTP Packet Header Extension. Taken from [39], this figure provides further detail on the RTP header extension mechanism. For this custom-built video streaming application, the extension is used to embed a timestamp that indicates when a packet is sent for network transmission. This timestamp enables the system controller to calculate the end-to-end delay. The length field is set to a value of one, which specifies a single 32-bit word in the extension payload..... 47

Figure 3.14: System Controller Graphical User Interface. Key components of the graphical user interface (GUI). The primary panel renders the incoming video stream. In the middle, a real-time analytics panel displays live plots of performance metrics. The control bar at the bottom contains buttons to play, pause, and stop the video stream, as well as to select the metric being visualized. 49

Figure 3.15: Network Interfaces used to Communicate with the System Controller. The devices depicted in this figure are network interfaces connected to the processing platforms of

the sensing nodes via USB. The top image shows the Wi-Fi device TP-Link Archer T2U Nano. To the bottom left is the Quectel RM530N-GL mmWave Kit, and to the right is the Quectel RM530N-GL configured with a sub 6GHz USB module. 52

Figure 3.16: Quectel RM530N-GL 5G Cellular Module. This device enables the sensing node cellular communication capability for both 4G LTE 1900MHz and 5G mmWave 39GHz bands. The module interfaces with a M.2 to USB carrier board to communication with the host system. It also connects with sub-6 and mmWave antenna via IPX coax cables. 53

Figure 4.1: Baseline Network Configuration. This diagram details the network configuration used as a baseline for comparing wireless configurations and processing platforms. It utilizes the device's internal network interface known as loopback. The benefit of this configuration is that it can negate the constricting effects of a wireless network. Furthermore, it enables us to isolate the devices processing power for an apples-to-apples comparison between devices..... 58

Figure 4.2: Wi-Fi 6 Network Configuration. The physical setup of the Wi-Fi network configuration is detailed in this diagram, which is organized into three distinct sections. The sensing node system is shown on the far left, positioned on a tabletop inside the AT&T 5G Lab at UTK. The middle section depicts the Wi-Fi router and modem on a separate tabletop, acting as an intermediary between the sensing node and the third section, the MEC Zone. The sensing node transmits its video stream to the Wi-Fi router over an IEEE 802.11ax 5GHz wireless channel with a 1 Gbps bandwidth. Upon receiving the data, the router forwards the video packets to the system controller, located in the MEC Zone, via a 10 Gbps fiber optic cable..... 59

Figure 4.3: Wi-Fi 6 Ericsson CradlePoint Router. This wireless gateway router operates as the access point for the sensing nodes and system controller. It enables Wi-Fi 6 5GHz communication between experimental devices. 60

Figure 4.4: Cellular Network Configuration. Both the 4G and 5G wireless modes utilize this setup, however, for the former, the mmWave antenna are disengaged to prohibit use of 39GHz. 61

Figure 4.5: AT&T 5G Lab Antenna and Radio Units. These units enable the cellular service to the user equipment (UE) operating in the lab. For this thesis, these UE are literally the sensing nodes which capture and stream video to the MEC resources within the cell site. The 4G LTE radio on the left operates at a frequency of 1900MHz and utilizes a massive input massive

output (MIMO). The 5G mmWave radio on the left operates at a frequency of 39GHz and utilizes a massive input massive output (MIMO). Both units forward their upstream data to the cell sites baseband units via a coaxial cable. 61

Figure 4.6: Processing Platform and 4G Cellular Configuration. This figure depicts three processing platforms, namely the Radxa Rock 5B on the left and Jetson Orin NX on the right, connected via USB 3.2 to a Quectel RM530N-GL 4G Cellular module. These devices compose the edge sensing node's processing platform. Similar configurations are assembled for wireless modes Wi-Fi 6 and 5G+ mmwave. 62

Figure 5.1: Comparative Analysis of Video Streaming Performance (RADXA). This figure presents the performance of the Radxa Rock 5B across three key metrics: bitrate, latency, and framerate. The trials were conducted under two variable conditions. The first is the network configuration, comparing a cellular 4G AT&T network (1900 MHz band) with a loopback configuration (LB), which uses the device's internal network interface. The second variable is the application of on-device processing, specifically facial feature extraction using the RetinaFace model. Each plot's legend includes the global average of the raw time series. 66

Figure 5.2: Comparative Analysis of Video Streaming Performance (JETSON). This figure presents the performance of the NVIDIA Jetson Orin NX across three key metrics: bitrate, latency, and framerate. The trials were conducted under two variable conditions. The first is the network configuration, comparing a cellular 4G AT&T network (1900 MHz band) with a loopback configuration (LB), which uses the device's internal network interface. The second variable is the application of on-device processing, specifically facial feature extraction using the RetinaFace model. Each plot's legend includes the global average of the raw time series. 66

Figure 5.3: Probability Distribution of Received Bitrate (RADXA). The plots displayed here summarize the distribution of the video stream's received bitrate for the Radxa Rock 5B. They highlight two primary findings. The scenarios without object detection consistently achieve a bitrate ~4.5 times greater than that with detection enabled, and enabling detection introduces significant performance variability, as shown by the much wider spread of the PDFs. The minimal difference between the 4G and loopback traces suggests that, for this application, network conditions were not the primary performance bottleneck. 67

Figure 5.4: Probability Distribution of Received Bitrate (JETSON). The plots displayed here summarize the distribution of the video stream's received bitrate for the NVIDIA Jetson

Orin NX. Like the Radxa device, traces captured from the tests without detection enabled achieved a increase in bitrate two times greater than that with detection enabled. However, there was approximately a 1 Mbps increase when transmitting across the loopback interface with detection enabled. Furthermore, the 95th percentile was 1.25 times greater than that using 4G. . 68

Figure 5.5: Probability Distribution of Framerate (RADXA). These plots display the PDF and CDF of framerate seen by the system controller when transmitted from the Radxa Rock 5B. A similar trend continues of greater bitrates for tests without detection enabled. For these traces, there was a ~4.3 times increase between detection enabled vs. disabled. 68

Figure 5.6: Probability Distribution of Framerate (JETSON). These plots display the PDF and CDF of framerate seen by the system controller when transmitted from the NVIDIA Jetson Orin NX. A similar trend continues of greater bitrates for tests without detection enabled. For these traces, there was a ~2-3 times increase between detection enabled and disabled. 69

Figure 5.7: Probability Distribution of Network Latency (RADXA). The PDF and CDF of the end-to-end latency between the Radxa Rock 5B and system controller give further insight into the performance of this configuration. Two key findings can be extracted. The first surrounds the network link, where 4G performance suffers greatly. Not only is there nearly a 100x increase in latency, but there is also significant variation. The second key finding can be seen by comparing the cases with detection enabled to those without. Enabling detection results in a decrease of variation, namely over a two times greater deviation for the 4G case and 1.5x greater for the loopback case. 70

Figure 5.8: Probability Distribution of Network Latency (JETSON). The PDF and CDF of the end-to-end latency between the NVIDIA Jetson Orin NX and system controller give similar insight into the streaming performance as the Radxa device. Again, the latency is much more consistent and lower than the 4G cases, and the deviation is significantly increased when disabling detection. 70

Figure 5.9: Raw Streaming Performance Data Across Wireless Technologies (Radxa/H.264). This figure displays a post analysis of the raw metrics logged by the benchmarking application run on the Radxa Rock 5B with the H.264 encoder enabled. It includes plots of bitrate, latency, frame rate, packet loss rate, and jitter from tests run across the wireless technologies 5G, 4G, and Wi-Fi 6. These results highlight key findings about performances achieved by the different wireless technologies. First, the bitrate remains relatively consistent

across technologies, varying by less than five percent on average. Second, Wi-Fi 6 achieves the lowest latency, while 4G exhibits the highest—nearly an order of magnitude greater. Third, the frame rate observed at the system controller remains largely stable across all wireless technologies, with only a minor decrease noted for 5G. Fourth, the packet loss plot shows that both 4G and Wi-Fi 6 maintain losses below one percent, whereas 5G experiences approximately four percent. Finally, the 4G test demonstrates the highest average jitter at roughly one millisecond, while 5G and Wi-Fi 6 deliver more consistent timing performance. 72

Figure 5.10: Raw Streaming Performance Data Across Wireless Technologies (Radxa/H.265). This figure displays a similar analysis as Figure 5.9, but the Radxa device was instead configured to use the H.265 encoder. These results reflect values nearly equal to those collected during the H.264 experiment; however, several differences can be observed. First, the average bitrate across the three wireless technologies is slightly lower, with the largest deviation not exceeding ten percent. Second, the ordering of latency remains consistent with that of the H.264 test, though both 4G and 5G exhibit modest increases by factors of 1.5 and 1.3, respectively. Third, the trend in packet loss rate persists, with 5G continuing to experience the highest loss, approximately 25% greater than in the H.264 test. Finally, jitter remains relatively low overall, though 4G demonstrates noticeably less stable delivery compared to 5G and Wi-Fi 6..... 73

Figure 5.11: Raw Streaming Performance Data Across Wireless Technologies (Jetson/H.264). This figure displays a similar analysis as Figure 5.9, but it includes results from tests using the NVIDIA Jetson Orin NX device configured to use the H.264 encoder. The more powerful NVIDIA device achieved roughly double the frame rate of the Radxa device but did not exceed an average bitrate of 5 Mbps. A different trend was observed for latency, with 5G consistently exhibiting the lowest values. Wi-Fi 6, which ranked second, showed a gradual linear increase in latency over the course of the test. Packet loss rates remained relatively low overall, with 5G demonstrating a slightly higher rate and Wi-Fi 6 maintaining the lowest. Jitter increased across all three wireless technologies compared to the Radxa device, with 4G once again producing the greatest variation in delivery times. 74

Figure 5.12: Raw Streaming Performance Data Across Wireless Technologies (Jetson/H.265). This figure displays a similar analysis as Figure 5.9, but it includes results from tests using the NVIDIA Jetson Orin NX device configured to use the H.265 encoder. The

NVIDIA device's H.265 encoder produced more than three times the bitrate of the H.264 encoder at the same frame rate. As in previous tests, 5G yielded the lowest latency, while Wi-Fi 6 and 4G exhibited values more than twice as high. The packet loss rate remained consistent with that observed in the H.264 experiment. The 4G network again demonstrated the highest average jitter among the three technologies. 75

Figure 5.13: Probability Distribution of Received Bitrate Across Wireless Technologies (Radxa). This figure presents the probability density function (PDF) and cumulative distribution function (CDF) of the received bitrate from tests conducted on the Radxa Rock 5B device using the 4G, 5G, and Wi-Fi 6 wireless technologies. The results highlight an overall increase in bitrate accompanied by greater variation in the H.264 experiments across all three technologies, with the 95th percentile being approximately 1.6 times higher and the standard deviation nearly three times greater. Overall, the average bitrate among the three technologies differed only marginally. 77

Figure 5.14: Probability Distribution of Received Bitrate Across Wireless Technologies (Jetson). This figure shows a similar analysis as Figure 5.13, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Unlike the results produced by the Radxa device, the 95th percentile for the H.265 encoder consistently reached values more than three times higher than those of the H.264 encoder. The distribution density around the median values in the H.264 tests was greater than that of the H.265 tests, with the largest difference observed for 4G, where the density was approximately 1.75 times higher. 78

Figure 5.15: Probability Distribution of Framerate Across Wireless Technologies (Radxa). This figure shows the probability density function and cumulative distribution function for the framerate produced by tests ran with the Radxa Rock 5B device across the wireless technologies 4G, 5G, and Wi-Fi 6. The frame rate values across wireless technologies differed by no more than two frames per second. For the 4G and 5G tests, the standard deviation observed in the H.264 configuration was slightly higher, approximately 60% greater than that of the H.265 configuration. 80

Figure 5.16: Probability Distribution of Framerate Across Wireless Technologies (Jetson). This figure shows a similar analysis as Figure 5.15, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Again, the 95th percentile remained almost equal across all tests. The lowest standard deviation was achieved by 5G at only twenty percent of the Wi-Fi 6 test. . 80

Figure 5.17: Probability Distribution of End-to-end Latency Across Wireless Technologies (Radxa). This figure shows the probability density function and cumulative distribution function for the end-to-end latency produced by tests ran with the Radxa Rock 5B device across the wireless technologies 4G, 5G, and Wi-Fi 6. Results varied considerably across the different configurations. Wi-Fi 6 demonstrated the best overall performance, with most latency values remaining around 5ms and a standard deviation of approximately 2ms. This performance is more than twelve times faster and ten times more stable than that of the slowest network, 4G..... 81

Figure 5.18: Probability Distribution of End-to-end Latency Across Wireless Technologies (Jetson). This figure shows a similar analysis as Figure 5.17, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Unlike the Radxa experiment, the NVIDIA device configuration experience the lowest latency and standard deviation with 5G enabled. The highest latency is observed in the Wi-Fi 6 results at eight times greater than 5G with a greater standard deviation by over thirteen times. 81

Figure 5.19: Probability Distribution of Packet Loss Rate Across Wireless Technologies (Radxa). This figure shows the probability density function and cumulative distribution function for the packet loss rate produced by tests ran with the Radxa Rock 5B device across the wireless technologies 4G, 5G, and Wi-Fi 6. The technology experiencing the greatest loss is 5G dropping at most 6% of packets with a variation reaching 1.3% throughout the experiment. Results for 4G and Wi-Fi 6 were relatively equal with a 95th percentile less than 0.5% and nearly zero variation. 83

Figure 5.20: Probability Distribution of Packet Loss Rate Across Wireless Technologies (Jetson). This figure shows a similar analysis as Figure 5.19, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Results from this device with its higher framerate vary widely compared to the Radxa experiments. 5G again demonstrated the greatest loss, reaching 1.5% using H.264, and 4G was a close second reaching 1% using H.264. Wi-Fi 6 maintained a consistent loss rate at near zero. 83

Figure 5.21: Probability Distribution of Average Jitter Across Wireless Technologies (Radxa). This figure shows the probability density function and cumulative distribution function for the average jitter produced by tests ran with the Radxa Rock 5B device across the wireless technologies 4G, 5G, and Wi-Fi 6. The greatest variation in packet delivery times was observed

in the 4G configuration, reaching greater than 2ms, which also exhibited a substantially higher standard deviation by more than double that of Wi-Fi 6 and 5G. Wi-Fi maintained the lowest jitter below 0.5ms and low standard deviation around 0.2ms..... 84

Figure 5.22: Probability Distribution of Average Jitter Across Wireless Technologies (Jetson). This figure shows a similar analysis as Figure 5.21, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Like the Radxa results, 4G experienced the most jitter reaching values above 4ms with greater standard deviation. Wi-Fi 6 and 5G networks performed relatively equally, with H.264 results reaching above 1ms. 85

Figure 5.23: Screenshot of Application's Live Dynamic Plot During Traffic Congestion Impairment (4G/Radxa). The plots shown here are screenshots captured from the benchmarking application during operation over the 4G network under an induced network impairment. They display real-time measurements of bitrate, frame rate, latency, jitter, effective delivery rate, and packet loss rate while competing traffic was active during the test. The gray dashed lines mark the moment when network congestion was introduced. The results reveal a clear change in performance metrics following the onset of impairment: both bitrate and frame rate begin decreasing linearly, latency increases to more than twice its pre-impairment value, and packet loss rate rises slightly at the point of congestion activation..... 87

Figure 5.24: Screenshot of Application's Live Dynamic Plot During Traffic Congestion Impairment (WiFi/Radxa). This figure shows similar plots as Figure 5.23, but it was taken from an experiment with Wi-Fi 6 configured as the network. A similar trend is observed compared to the 4G test. Both bitrate and frame rate exhibit a linear decline, while latency rises sharply to more than four times its pre-impairment value. Packet loss rate also increases at the moment the impairment is activated. 88

CHAPTER ONE
INTRODUCTION

The work presented in this thesis attempts to benchmark and analyze edge computing systems using various processing resources, wireless networking technologies, and operating conditions. Methods for a unified benchmarking framework are discussed and results from several tests are analyzed. Additionally, the strengths and weaknesses of each configuration are analyzed, and a discussion of design considerations is given.

Motivation

The development of edge computing systems began in the early 1990s, a direct response to the growing emphasis on highly interactive multimedia [1]. As technology advanced, the effort to improve massive data flow and responsiveness at the network edge became a core focus, giving rise to solutions like content delivery networks and Multi-Access Edge Computing (MEC) infrastructure in cellular networks [2]. Today, this field has matured significantly, with edge technologies being deployed alongside critical infrastructure to ensure public safety and economic stability [3]. One example application is smart agriculture, seen in **Figure 1.1**, where a massive amount of interconnected telemetry and control devices share real-time information. Similar modern applications such as intelligent transportation systems and telemedicine rely on the high-bitrate and low-latency capabilities that edge computing provides [4, 5]. This convergence of demand and recent advancements in wireless communication and distributed computation has fundamentally shifted the research and development focus toward achieving real-time performance at the network edge [6].

This thesis examines edge computing in contexts that include critical infrastructure, where reliability and performance are closely tied to safety considerations. One such context is autonomous driving, seen in **Figure 1.2**, where vehicle-to-vehicle (V2V) and vehicle-to-everything (V2X) communication must exhibit extreme reliability and responsiveness [7]. These applications require the edge to process large volumes of sensory input and make decisions within milliseconds—tasks that traditional cloud architectures, with their higher latency and bandwidth bottlenecks, are poorly suited to handle [7].

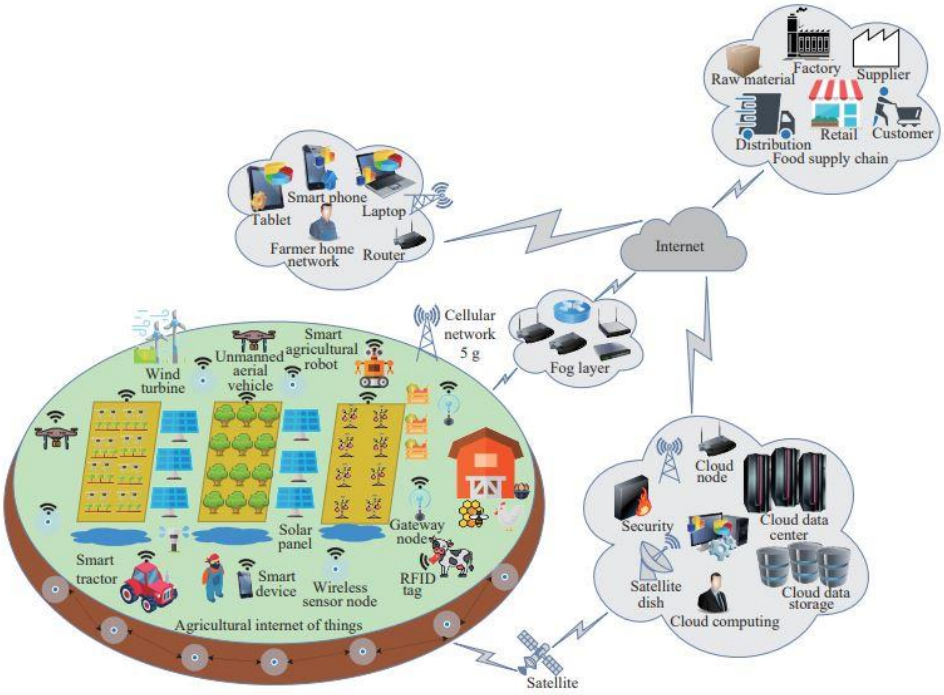

Figure 1.1: IoT-connected smart agriculture sensors enable the IoT. This example smart farm taken from [8] demonstrates the communications scalability requirement of modern applications.

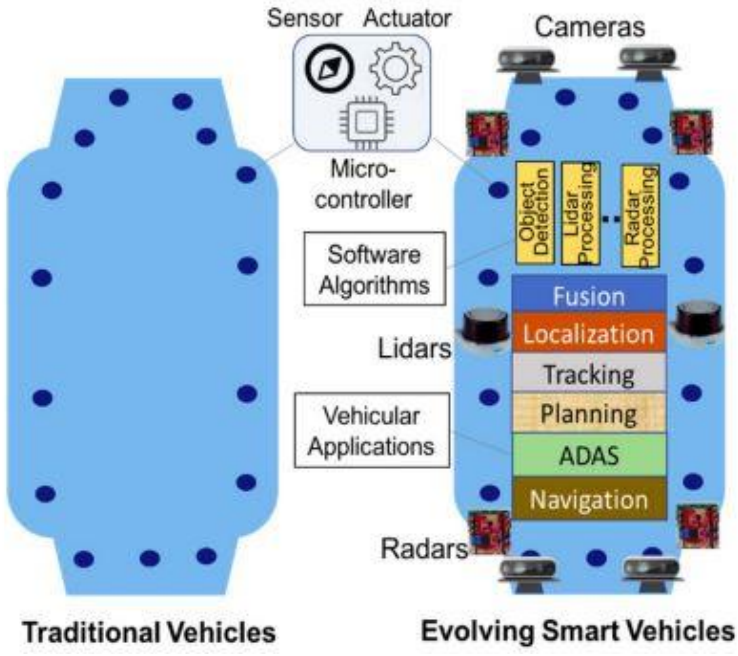

Figure 1.2: Evolution of vehicular sensors, computing tasks and applications from traditional to connected and autonomous vehicles. These large volume real-time advanced vehicular applications, taken from [9], illustrate the sheer number of sensing functions integrated with vehicles.

Experimental setups of this work zero in on the final hop of edge computing systems: the point at which latency-sensitive applications physically reside on a sensing node, processing platform, or user device. These edge environments, while highly dynamic and heterogeneous, offer the closest proximity to the data source and thus the greatest potential for ultra-low-latency computing. Understanding and improving the reliability of this last-mile communication is vital, especially as modern applications demand stringent Quality-of-Service (QoS) and Quality-of-Experience (QoE) guarantees [10].

Computing and communication devices used in the experimentation of this thesis leverage low-cost, off-the-shelf sensing systems. These inexpensive platforms allow for widespread experimentation, rapid prototyping, and custom solutions tailored to specific applications. Yet, their low cost comes with trade-offs in terms of processing power, memory constraints, and software flexibility, which are included in the final analysis.

The central aim of this thesis is to construct and deploy a benchmarking platform that can simulate real-world network scenarios involving Wi-Fi, 4G LTE, and 5G mmWave. The goal is to study how various edge applications, particularly multimedia streaming and real-time video analytics, respond to fluctuating network conditions when deployed at the edge. By coupling performance monitoring with system-level data logging and visualization tools, the work aspires to contribute a flexible, modular framework for evaluating edge systems under operational constraints that mirror deployment environments.

In essence, this research is driven by the need to better characterize the performance of next-generation edge computing platforms, identify their limitations, and guide the design of more resilient and responsive architectures. The ultimate objective is to ensure that future edge systems can meet the stringent demands of public safety, automation, and immersive user experiences.

Pipeline of Thesis

The research methodology employs a pipeline, depicted in **Figure 1.3**, designed to organize the workflow and aid in the overall comprehension of its main goals. It is divided into two parts: the "Analysis Phase" and the "Application Phase." In the "Analysis Phase," the operating conditions of the benchmarking system are considered. This includes the hardware and software platforms used for the critical service, the network technology linking the client and server, and the cyber-physical space where data will propagate. Depending on the characteristics

of each system's configuration, the performance will vary in the "Application Phase." Here, a common benchmark task will be executed on the system and performance metrics will be gathered. These real-world metrics are then used to quantitatively grade and rank a configuration's ability to complete the task.

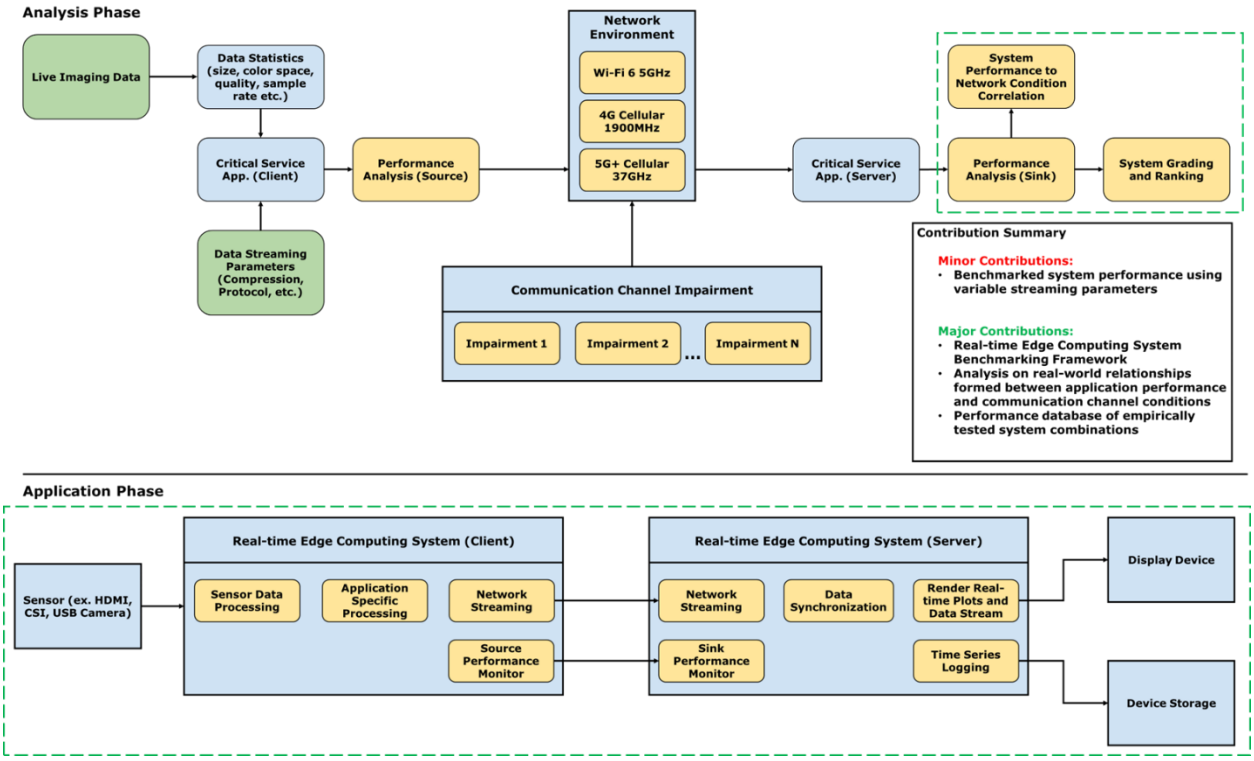

Figure 1.3 Thesis Pipeline of data capture, edge processing, network transmission, impairment injection, multimedia streaming, and analysis. The thesis is divided into two phases, Analysis and Application. The Analysis Phase details the variable conditions of the edge computing system along with the methods for interpreting the resulting data. In this phase, the live data source, processing platform, wireless network technology, channel impairment, and layer 4 protocols are varied to determine their impact on system performance. The Application Phase consists of the cyber-physical system which streams data from client to server and benchmarks the data stream performance. It will also offer real-time visualization of the benchmarking metrics.

Thesis Contributions

There are four contributions made by this thesis. They are categorized as ‘major’ and ‘minor’ to distinguish their level of depth and novelty.

Contribution 1 (Major)

This work's primary contribution is a novel benchmarking framework designed to quantify the performance of specific system configurations. It comprises three interchangeable components: sensing, processing, and communication. Within a defined testbed, this framework

enables the collection of real-world data, its processing with application-specific algorithms, and transmission over a specified physical environment. This allows for the determination of a system's practical capabilities, particularly under demanding processing and communication loads.

A key aspect of this framework is its critical service model. This model defines system requirements, aiming for the highest possible bitrate and lowest latency, while also considering acceptable data loss, practical distance limitations, and resource consumption. These constraints collectively form the benchmark's testing criteria. Furthermore, the framework exclusively focuses on edge paradigms, meaning all device communication occurs over local-area networks (LANs), without reliance on wide area networks.

Contribution 2 (Major)

This research's second contribution is the quantification of an application's performance subject to a dynamic network environment, ranging from ideal to detrimental conditions. By introducing impairments to both the physical radio channel and the digital network, we demonstrate and measure a predictable impact on bitrate, latency, and loss as experienced by a sensitive edge application, all within a controlled benchmarking framework.

Contribution 3 (Major)

A third significant contribution of this research is the compilation of a comprehensive database of benchmark results derived from extensive empirical tests. These tests were systematically conducted across a variety of distinct system configurations (e.g., different hardware specifications, processing capabilities) and diverse network environments (e.g., varying wireless channel conditions, injected digital impairments, different traffic loads). This robust database provides a detailed survey of the expected performance for typical device combinations under specified network conditions. It offers valuable insights into how factors like processing power, wireless signal quality, and network congestion directly influence critical application metrics such as bitrate, latency, and data loss in real-world scenarios. This resource serves as a foundation for understanding performance capabilities and informing future system design in edge computing paradigms.

Contribution 4 (Minor)

A final, minor contribution of this work involves subtle software modifications on the tested devices. These modifications are specifically designed to push the performance

capabilities of the experimental setups beyond their baseline. This includes implementing refinements and optimizations to the available streaming protocols and compression techniques. By doing so, this enhancement significantly increases the framework's applicability to real-world scenarios by reflecting more aggressively optimized systems. Consequently, it enables a more comprehensive and nuanced analysis of the system's true performance limits under various conditions.

Synopsis

This thesis is structured to guide the reader from the foundational concepts of edge computing to the practical implementation, evaluation, and analysis of a novel benchmarking framework. Each chapter builds upon the last, collectively addressing the core motivation of quantifying the real-world performance of edge systems for critical applications.

In Chapter Two, the academic and technical context for this work is established. It delves into the foundational principles of edge computing, its architectural models, and its crucial role in critical infrastructure applications. The chapter then surveys the modern wireless technologies central to this study—Wi-Fi 6, 4G LTE, and 5G—and examines existing methods for performance benchmarking. By reviewing current streaming protocols, compression techniques, and the known effects of network impairments, this chapter identifies the research gap that the proposed benchmarking framework is designed to fill.

In Chapter Three, the primary contribution of this thesis, the design and construction of the unified benchmarking framework, is presented. It details the high-level architecture, which consists of a sensing node, a network gateway, and a system controller. This chapter provides a comprehensive breakdown of the hardware platforms, including the single-board computers and wireless modems used, and the software components, with a focus on the GStreamer multimedia framework and custom modifications to the Real-time Transport Protocol (RTP) for precise metric collection. The implementation of the application-specific processing task, facial feature extraction, is also described, grounding the benchmark in a realistic edge AI workload.

In Chapter Four, the scientific approach used to evaluate the system is outlined. It describes the specific physical and digital setups for each experiment, including the baseline (loopback), Wi-Fi 6, 4G LTE, and 5G configurations. This chapter defines the key performance metrics—such as bitrate, latency, and frame rate—that are captured to quantify system performance. Furthermore, it details the procedures for conducting the experiments and the data

analysis techniques that will be used to interpret the results, ensuring the study's reproducibility and scientific rigor.

In Chapter Five, the empirical findings from the extensive experimental trials are presented. The results are analyzed across several dimensions, comparing performance between different processing platforms, wireless technologies, and application parameters like the enabling of on-device processing. This chapter directly addresses the thesis's second and third major contributions by quantifying how network conditions and processing loads impact application performance and by providing a robust database of benchmark results. The discussion section synthesizes these findings, highlighting key performance trade-offs and their practical implications for deploying critical services at the wireless edge.

In Chapter Six, the research is summarized, the key findings are reiterated, and the fulfillment of the thesis's contributions is considered. It provides a final synthesis of the work's significance for engineers and researchers in the field. The chapter concludes by offering recommendations for designing and deploying resilient edge computing systems and suggests promising directions for future work that can build upon the framework and findings presented in this thesis.

Conclusion

Chapter One established the foundation of this thesis by defining the central research problem, motivating its real-world importance, and outlining the methodological and technical trajectory that follows. It identified the growing challenge of sustaining low-latency, high-reliability communication for modern edge applications—such as public safety, automation, and intelligent transportation—operating over inherently variable wireless technologies including Wi-Fi 6, 4G LTE, and 5G mmWave. Because these environments exhibit temporal and spatial variability that cannot be adequately modeled through static assumptions, the chapter argued for a real-world, time-synchronized benchmarking framework capable of measuring both application-level performance and underlying network behavior in real time.

To address this challenge, Chapter One introduced a structured research pipeline composed of two phases: Analysis, which defines the experimental parameters, variable factors, and interpretation methods; and Application, which executes controlled workloads under synchronized observation. This dual-phase approach provides both methodological rigor and

practical flexibility—ensuring that results are comparable across devices, network modes, and configurations while remaining representative of real-world operational conditions.

The chapter also articulated the thesis's major contributions: a unified benchmarking framework integrating sensing, processing, and communication components for realistic edge evaluation; a quantitative characterization of how wireless and computational factors jointly affect throughput, latency, and stability; and a curated empirical dataset spanning diverse platform and RAT combinations to inform future design decisions. Supporting these are incremental innovations in streaming protocol design, such as custom RTP timestamp extensions and codec-level optimizations that enhance timing precision and observability.

Finally, Chapter One distilled a set of methodological commitments that underpin the entire thesis: precise in-band timing for one-way delay measurement, synchronized cross-layer observability, consistent workload and RAT variation for controlled comparison, and impairment-aware testing to capture stability under realistic stress. These guiding principles ensure that all subsequent claims rest on synchronized, verifiable evidence rather than abstract modeling.

In summary, Chapter One defines the problem space, establishes the research scope, and commits to a reproducible, cross-layer benchmarking approach that connects edge application behavior with wireless network dynamics. With this conceptual groundwork laid, the thesis proceeds in Chapter Two to situate these aims within the broader technical and academic context—reviewing prior work in edge architectures, wireless technologies, and benchmarking methodologies that inform the system design and experimental framework developed in later chapters.

CHAPTER TWO
LITERATURE REVIEW

Edge computing has emerged as a transformative paradigm for latency-sensitive and reliability dependent critical applications, bringing computation closer to the data source to minimize transmission delays and improve responsiveness [1, 6]. When deployed over modern wireless technologies such as Wi-Fi 6, 4G LTE, and 5G these systems promise unprecedented mobility and throughput, yet face real-world variability that can undermine application performance [11-13]. High-performance multimedia communications, particularly real-time sensor processing and streaming, push the limits of both edge devices and wireless links, demanding sustained bitrate, low latency, and consistent frame delivery even under fluctuating conditions [14, 15]. Performance benchmarking and real-time systems research provide the methodological backbone for quantifying these capabilities, enabling the evaluation on the practical limits of off-the-shelf hardware and optimized protocols for field deployment [16, 17]. In this context, this thesis develops and applies a portable benchmarking framework to characterize how consumer-grade edge platforms meet the stringent requirements of critical infrastructure applications, capturing both application-level quality metrics and link-layer network conditions to inform robust system design.

Foundations of Edge Computing in Critical Infrastructure

To understand the system-level trade-offs this work seeks to evaluate, it is first necessary to examine the foundational concepts and design principles of edge computing, with particular attention to its role in latency-critical and high-reliability applications such as automation, manufacturing, and security. This discussion will begin with the definitions and architectural philosophy of edge computing, then it explores how these ideas are applied to real-world problems, and finally it will conclude with a brief introduction to Multi-access Edge Computing (MEC).

Definition and Architectural Models of Edge Computing

In contemporary research, such as the work by Kumari et al., discussions of edge computing are almost always framed in relation to the cloud and the Internet of Things (IoT) [18]. Their work describes the cloud as representing a vast pool of centralized computing and storage resources, while the IoT encompasses billions of often small, resource-constrained

devices connected to the internet. Edge computing bridges the gap between these two domains by placing processing and storage capabilities closer to where data is generated.

As seen in **Figure 2.1**, deploying edge resources typically requires inserting an additional layer, extending or modifying networking infrastructure to position compute nodes at or near access points, gateways, or other aggregation layers. This architectural shift reduces end-to-end propagation delays, alleviates congestion within the network core, and minimizes unnecessary backhaul traffic. As a result, edge computing enables more responsive, interactive, and time-sensitive communication between IoT devices and distributed compute resources.

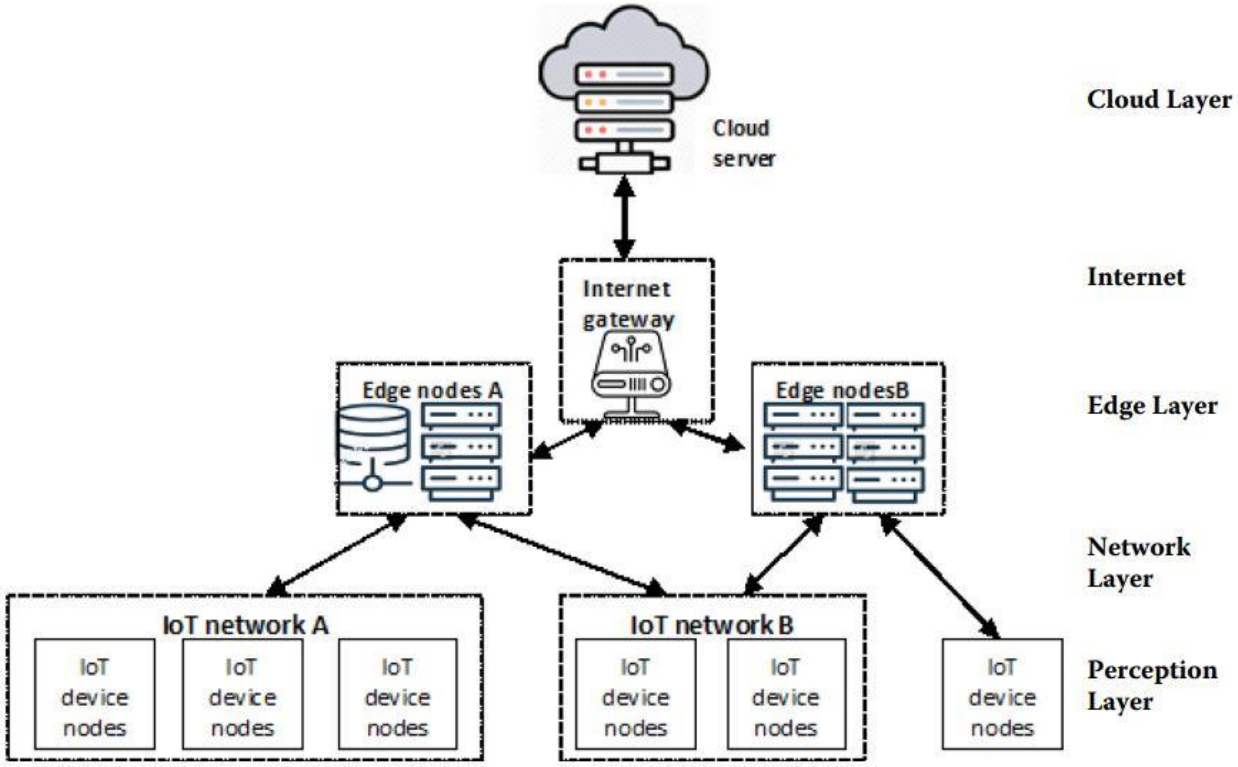

Figure 2.1: High-level architecture of an edge computing system. Taken from [18], this diagram depicts the typical architecture for edge computing. It begins with the cost-effective, special purpose edge devices in the perception layer. Then data travels upstream through the network layer to the edge compute resources where higher-capacity processing exists. If needed data can travel further into the internet to access cloud resources.

The benchmarking framework proposed in this thesis concentrates specifically on the "final hop" of this paradigm, the critical link between IoT devices and their proximate edge resources, where latency, reliability, and bandwidth constraints most directly influence application performance. This leads to the importance for the deployment of rigorous benchmarks on critical infrastructure and applications at the edge.

Role in Critical Applications

Applications highlighted in the motivation of this thesis, namely smart agriculture, intelligent transportation, and telemedicine, depend heavily on real-time communication and computation to ensure timely responses, particularly in contexts involving public safety. As these critical domains continue to expand and integrate more deeply into society, vital infrastructure is increasingly designed and deployed to service their stringent performance requirements, this can be seen in the Digi International Inc. blog post on AT&T and Digi deployment of critical transportation infrastructure in NYC [19, 20]. As presented by Lorio et al., the study of edge computing must move beyond the base architectural considerations to examine how effectively it delivers low-latency, high-reliability performance under real-world operating conditions [21].

One example of edge computing applied to public safety is the UAV-based edge architecture for disaster scenarios presented by Kaleem et al [22]. This work demonstrates how edge-enabled UAV networks can provide resilient, low-latency communications when traditional terrestrial radio systems are unavailable or compromised. Their proposed design incorporates UAV ad-hoc backhaul and cloudlet nodes, achieving 20 percent lower delay and reduced power consumption compared to conventional architectures, highlighting the potential of edge computing to enhance emergency response capabilities.

Taken together, these examples illustrate how edge computing supports mission-critical domains where communication delays or system failures can have severe consequences. Whether through transportation systems that manage urban mobility, telemedicine platforms that deliver remote healthcare, or emergency response networks augmented by UAV-based infrastructure, a common thread emerges. Edge resources serve to guarantee responsiveness, reliability, and resilience when centralized solutions alone are insufficient. More broadly, critical applications of edge computing can be characterized into categories such as public safety and emergency management, healthcare and telemedicine, intelligent transportation and industrial automation, each imposing unique requirements on latency, reliability, and scalability.

Real-time communication in critical applications is challenging because these systems combine unpredictability, strict reliability requirements, and safety-critical consequences. For example, intelligent transportation systems must process data from sensors and V2X communications within milliseconds to prevent accidents in highly dynamic environments. Telemedicine applications like remote surgery rely on uninterrupted high-resolution video and

haptic feedback, where even brief delays can jeopardize patient safety. Smart agriculture systems depend on large-scale, variable sensor traffic, where delayed or lost data can disrupt time-sensitive interventions such as irrigation or pest control. Across these domains, the common thread is that communication delays or failures are unacceptable, yet networks are inherently prone to congestion, interference, and mobility-related disruptions. This gap between stringent application demands and unreliable network conditions makes ensuring real-time communication a core challenge and a key focus of edge computing research.

Edge computing research in critical applications must go beyond architectural design to demonstrate its ability to deliver low-latency, high-reliability performance under real-world conditions. For example, UAV-based edge networks for disaster response highlight how edge computing enhances public safety by providing resilient, timely communication when traditional systems are unavailable or compromised. More broadly, edge computing is essential across mission-critical domains, enabling responsiveness, reliability, and resilience that centralized systems alone cannot achieve. The difficulty of guaranteeing real-time communication in these applications underscores why meeting these stringent requirements is a central focus of edge computing research.

Multi-access Edge Computing

The cellular network utilized in the experimentation of this thesis is a private Multi-access Edge Computing (MEC) cell cite installed at the University of Tennessee (UTK); therefore, it is imperative to understand the inner workings of its architecture, the differences from traditional edge infrastructure, and its role in critical applications. MEC has emerged as a significant area of development within modern networking and distributed computing. It reflects the broader trend of relocating certain computational functions and data handling processes toward the periphery of communication networks. This approach aligns with efforts to improve responsiveness and adaptability in systems that operate across diverse network environments. MEC is now discussed in conjunction with advancements in wireless infrastructure, including 5G, as part of the continuing evolution of connected and intelligent network architectures. Within this architecture, the foundational layer consists of mobile nodes interconnected through the radio access network, forming the transmission backbone for edge-based services as seen in the framework diagram of **Figure 2.2**.

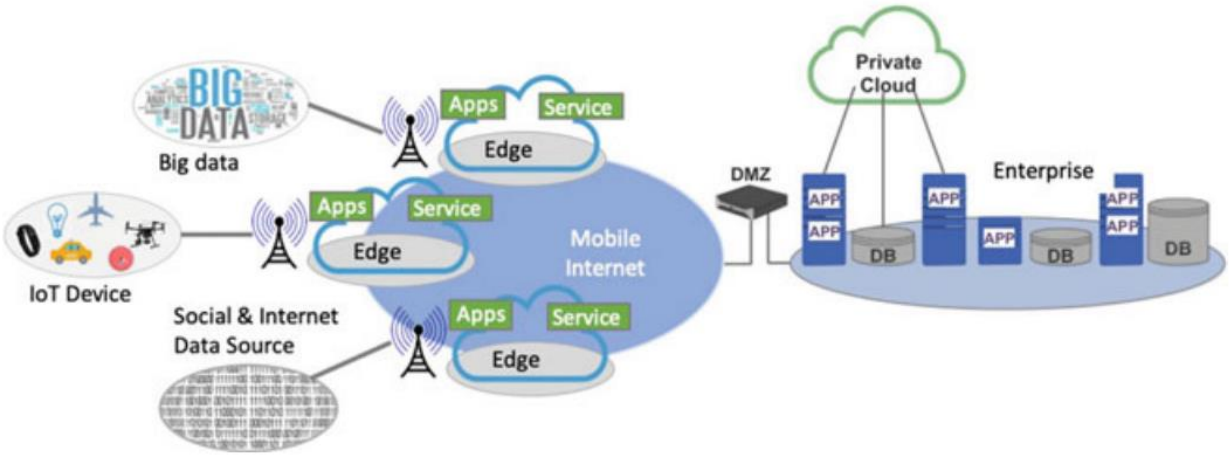

Figure 2.2: General MEC Framework. Taken from [23], this figure gives a general architecture for edge computing and highlights its connection to end devices and cloud infrastructure. The edge devices offer services and computing closer to the end users mitigating upstream traffic to cloud resources. Furthermore, domains like IoT devices and sources of big data can allocate edge resources to get a speedup in response times over conventional web services in a large data center.

Definition of Multi-Access Edge Computing (MEC)

This novel network architecture, as defined in [24] places computing, storage, and networking resources at the edge of the network, closer to end users and devices. It enables data processing and service delivery within or near the radio access network, reducing the distance between data generation and computation. Within the broader context of edge and cloud computing, MEC functions as an intermediary layer that supports localized processing, low-latency communication, and efficient integration between user equipment and centralized cloud services.

The Multi-access Edge Computing (MEC) reference architecture defines the key functional components of the MEC system and the reference points that govern their interactions. At its core, the system consists of MEC hosts and MEC management components, which together enable the deployment, orchestration, and operation of MEC applications within an operator network. Each MEC host includes a MEC platform and a virtualization infrastructure that supplies the compute, storage, and networking resources required to execute MEC applications. These applications operate on the virtualization layer and can provide or consume MEC services, depending on their configuration and purpose.

The MEC management framework operates on two levels, as shown in **Figure 2.3**. At the system level, the MEC orchestrator oversees the entire MEC environment, managing resources,

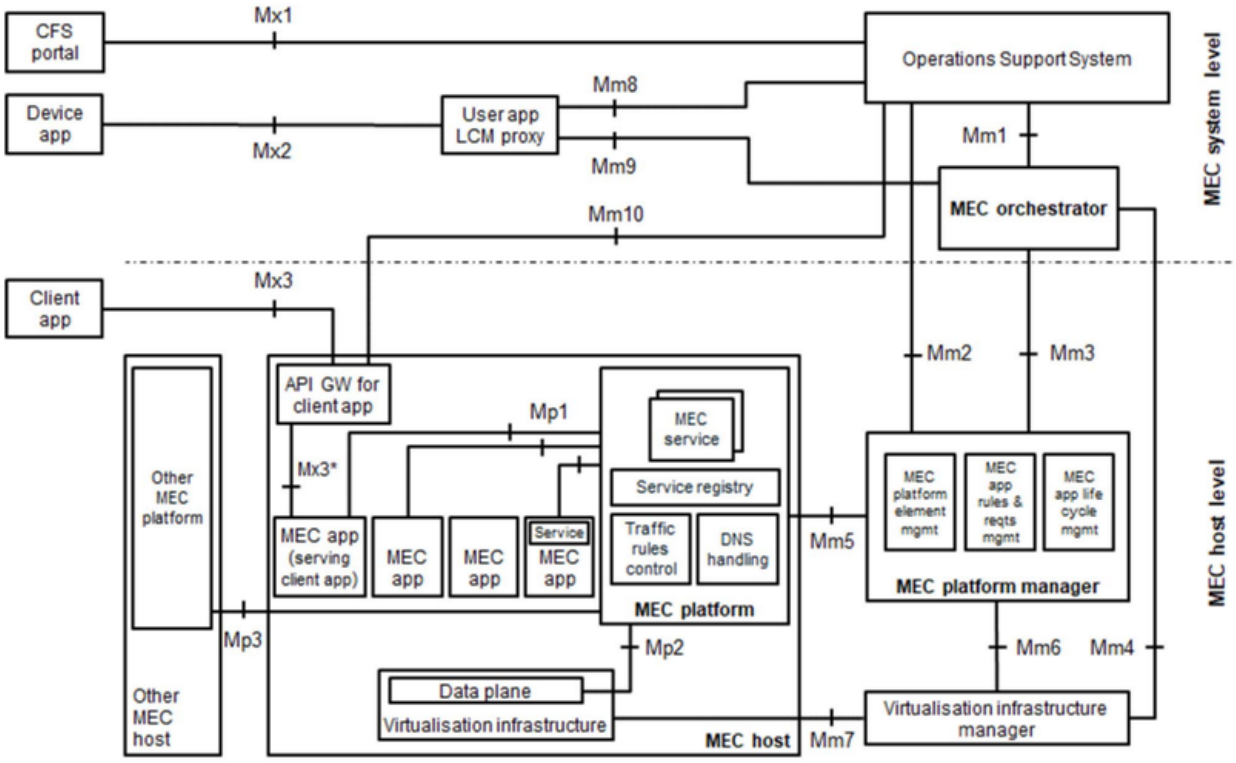

Figure 2.3: Multi-access Edge System Reference Architecture. Taken from [24], this diagram outlines the functional components and their interaction points of the MEC design.

coordinating service deployment, and ensuring policy compliance across hosts. At the host level, the MEC platform manager and virtualization infrastructure manager handle local operations, including application lifecycle management and platform-specific functionality. External entities such as client applications, device applications, and Customer Facing Service (CFS) portals connect to the MEC system through well-defined reference points: Mp (MEC platform functions), Mm (management interfaces), and Mx (external interfaces). An API gateway regulates access between client applications and MEC-hosted services, ensuring secure and controlled communication.

Overall, the MEC system functions as a distributed computing environment that brings cloud capabilities closer to users, supporting low-latency, high-performance services within mobile and edge networks. Through the coordinated interaction of its hosts, management components, and external interfaces, MEC provides a scalable framework for deploying and managing edge applications that enhance responsiveness, efficiency, and service quality across diverse network domains. These performance enhancing advantages are the reason for integration of MEC with critical infrastructure.

MEC Integration with Critical Infrastructure

With the promise of low latency and high availability, MEC has emerged as a strong candidate for integration with critical infrastructure systems such as disaster response and intelligent surveillance. Rigorous testing and real-world validation are essential for these applications to ensure public safety and operational reliability. The unified benchmarking framework developed in this thesis represents a step toward addressing this need. Using the UTK–AT&T 5G MEC Lab as a testing environment, the framework provides direct access to technologies currently deployed in communication infrastructure and industry. This section highlights the major benefits and challenges that must be considered when designing networks and deploying applications in such environments. Two representative studies are presented that demonstrate the role of MEC in applications demanding high computational performance, rapid response, and reliability.

The first of these studies, conducted by Zhang et al., explores methods for reducing energy consumption in UAV-assisted disaster rescue operations through the integration of MEC resources and advanced algorithms [25]. From both communication and computation perspectives, the authors employed artificial intelligence models to optimize path planning and offloading strategies, thereby reducing energy use and improving responsiveness. Their simulation results demonstrated measurable improvements in both response time and energy efficiency, supporting the potential of MEC-enabled architectures in time-sensitive, resource-constrained scenarios.

Research by Modal et al. addresses resource limitations in intelligent surveillance systems by leveraging MEC to enhance computational capabilities and reduce dependency on remote cloud resources [26]. The study directly compared a MEC-based framework with a traditional cloud-based configuration and found that MEC significantly improves cost-effectiveness and energy efficiency through localized offloading strategies. The proposed system outperformed its cloud-based counterpart across multiple performance indicators, including latency, execution cost, network utilization, execution time, and overall energy consumption.

Together, these studies demonstrate the tangible advantages of integrating MEC into applications that operate under stringent constraints for robustness, responsiveness, and performance. Such integration enables the development of next-generation critical systems that can dynamically balance computational load, improve availability, and deliver near real-time

results. However, these benefits are accompanied by new challenges. MEC deployments require careful configuration of network and compute resources, along with robust cybersecurity mechanisms to protect sensitive operations. Furthermore, efficient resource management and intelligent task scheduling demand advanced algorithms capable of adapting to changing workloads and network conditions.

In summary, the studies reviewed in this section reinforce the central goals of this thesis—to establish a standardized benchmarking framework that can evaluate and compare the performance of edge and wireless technologies under real-world conditions. By aligning with the same principles of responsiveness, energy efficiency, and scalability observed in the reviewed works, the framework developed here provides a practical foundation for assessing MEC integration within critical and mission-dependent systems.

Modern Wireless Network Technologies for Real-Time Applications

Building upon this foundation, the next step is to review the capabilities and constraints of the wireless technologies, Wi-Fi 6, 4G LTE, and 5G NR, highlighting both their theoretical advantages and their real-world performance limitations. The survey of wireless networks in the context of Industrial Wireless Sensor and Actuator Networks (IWSAN) conducted by Seferagic et.al., presents key performance requirements of various industrial systems. A summary of these findings can be found in **Figure 2.4**.

As expressed in [27], the International Society of Automation (ISA) classifies industrial systems into six categories based on data urgency, ranging from highly critical safety systems demanding near-instantaneous, ultra-reliable action to non-critical information gathering systems. While wireless technology is suitable for many less-critical applications, the stringent ultra-low latency and high redundancy requirements of systems make them infeasible for coverage by long-range wireless solutions. Emerging wireless solutions offer much higher throughputs and lower latencies which could be utilized by real-time application with these stringent requirements.

Wi-Fi 6

The first wireless technology analyzed in the chapter is Wi-Fi 6 defined by IEEE 802.11ax standard. This standard represents the sixth generation of Wi-Fi technology and focuses primarily on boosting network efficiency and capacity rather than just peak speed, especially in dense, crowded environments. As defined in the standard [28], it achieves this by simultaneously

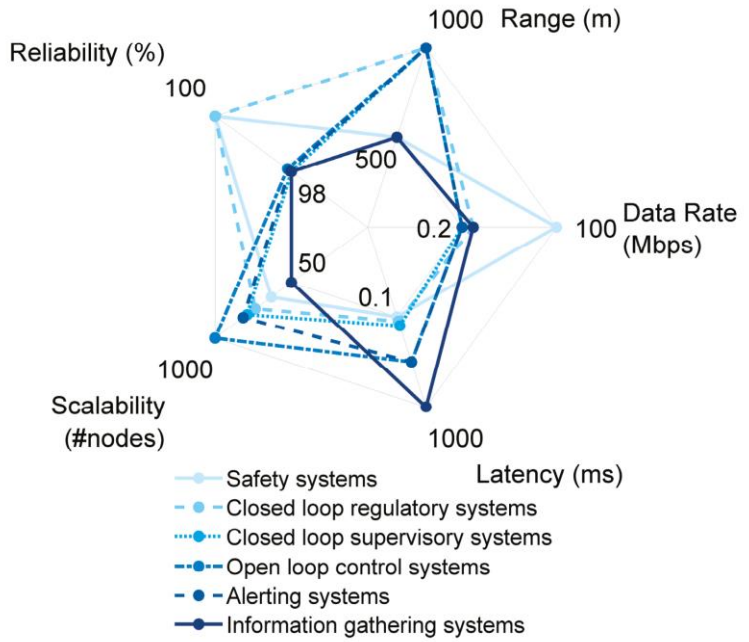

Figure 2.4: Industrial System Classes and their Performance Requirements. Taken from [27], this figure highlights the varying needs of critical applications. Safety systems, for example, rely heavily on low latency and high reliability, but they have relatively lower requirements for data rate and scalability.

modifying the Physical (PHY) Layer and the Medium Access Control (MAC) sublayer. The core technologies defining Wi-Fi 6 include Orthogonal Frequency Division Multiple Access (OFDMA), which divides channels to serve multiple clients concurrently in one transmission; enhanced Multi-User Multiple-Input Multiple-Output (MU-MIMO) for simultaneous uplink and downlink data handling; and 1024-QAM modulation for higher data density. Additionally, it features Target Wake Time (TWT) to dramatically improve battery life for mobile and IoT devices and BSS Coloring to reduce interference from neighboring networks, resulting in a more reliable and lower-latency experience for all connected users across the 2.4 GHz and 5 GHz.

Based on the capabilities defined by the IEEE 802.11ax (Wi-Fi 6) standard, this technology is highly relevant and effective for demanding real-time multimedia communication, low-latency, and high-bandwidth applications, particularly in dense user environments. While it does use 1024-QAM modulation to increase peak data density, its primary strength for quality of service (QoS) lies in enhancements to network efficiency and capacity. The adoption of Orthogonal Frequency Division Multiple Access (OFDMA) at the PHY and MAC layers allows the channel to be subdivided, enabling multiple users to transmit simultaneously within the same time-frequency resource block, which dramatically reduces medium access delay and improves

the determinism. Similarly, the simultaneous handling of uplink and downlink data via enhanced Multi-User Multiple-Input Multiple-Output (MU-MIMO) ensures that the aggregate high-bandwidth demands of concurrent users can be met without significant degradation in throughput per user. Features like BSS Coloring further contribute to improved latency and reliability by mitigating co-channel interference thereby positioning Wi-Fi 6 as a robust solution for next-generation multimedia services.

4G LTE

The 4G Long-Term Evolution (LTE) standard, defined primarily in 3GPP Release 8 and 9, fundamentally redefined cellular networks by introducing the Evolved Universal Terrestrial Radio Access Network (E-UTRAN) and the Evolved Packet Core (EPC), forming an all-IP, end-to-end packet-switched system [29]. As described in the standard, the core technology that defines the E-UTRAN's air interface is a new radio access scheme utilizing Orthogonal Frequency Division Multiple Access (OFDMA) for the downlink and Single-Carrier Frequency Division Multiple Access (SC-FDMA) for the uplink, both of which are highly efficient techniques that maximize spectrum utilization and significantly lower latency compared to 3G. Furthermore, the standard for LTE incorporates MIMO (Multiple-Input Multiple-Output) antenna technology to enhance spectral efficiency and robustly uses Adaptive Modulation and Coding (AMC) to adjust data rates based on channel conditions. At the network level, LTE operates in and interacts with the Physical Layer (PHY) and the Medium Access Control (MAC), Radio Link Control (RLC), and Packet Data Convergence Protocol (PDCP) sublayers within the data link layer of the E-UTRAN, all of which communicate with the higher-layer protocols of the EPC, stated to function as the core network by 3GPP Release 8.

The LTE air interface significantly boosted bandwidth through the utilization of Orthogonal Frequency Division Multiple Access (OFDMA) in the downlink and Single-Carrier Frequency Division Multiple Access (SC-FDMA) in the uplink, enabling higher spectral efficiency and peak data rates. Furthermore, the integration of MIMO (Multiple-Input Multiple-Output) antenna technology provided a substantial enhancement to capacity, allowing multiple data streams to be transmitted simultaneously, which is fundamental for supporting concurrent high-bandwidth users and multimedia traffic. While LTE's latency did not target the ultra-low latency levels required for true real-time mission-critical applications, its low-latency design, facilitated by a simplified network structure and the efficiency of its radio access schemes,

successfully enabled widespread adoption of high-quality mobile video streaming, HD voice (VoLTE), and general mobile broadband, effectively defining the first generation of mobile multimedia services.

5G NR

The 5G New Radio (NR) standard, which was first defined in 3GPP Release 15, represents a complete evolution from 4G LTE, designed to support three distinct service categories: Enhanced Mobile Broadband (eMBB), Massive Machine-Type Communications (mMTC), and Ultra-Reliable Low-Latency Communications (URLLC) [30]. As described in the standard, key defining technologies of 5G NR include the use of a more flexible Orthogonal Frequency Division Multiplexing (OFDM) waveform with scalable numerology (subcarrier spacing) to operate across a vastly wider spectrum, spanning from sub-1 GHz bands (Frequency Range 1, FR1) up to millimeter-wave (mmWave) frequencies (Frequency Range 2, FR2) that can reach tens of GHz. This is coupled with Massive Multiple-Input Multiple-Output (Massive MIMO) antenna systems and Intelligent Beamforming, which according to 3GPP Release 15, allow the base station (gNB or Next Generation Node B) to focus radio energy directly toward the user equipment (UE), drastically increasing spectral efficiency and network capacity.

Much like 4G LTE, the 5G NR radio access network operates within the Physical Layer (PHY), and the Medium Access Control (MAC), Radio Link Control (RLC), Packet Data Convergence Protocol (PDCP), and a new layer called the Service Data Adaptation Protocol (SDAP) within the data link layer [30]. These layers interact with the higher-layer Radio Resource Control (RRC) protocol, and the entire access stratum connects to the 5G Core (5GC) network, which is designed with a cloud-native, Service-Based Architecture (SBA) [30].

This technology offers a revolutionary leap in capabilities for real-time multimedia communication, low-latency, and high-bandwidth services, fundamentally transcending the performance limits of 4G LTE. Unlike LTE, which was optimized primarily for enhanced mobile broadband (eMBB), 5G NR was architecturally designed to explicitly support Ultra-Reliable Low-Latency Communications (URLLC), allowing it to achieve deterministic latencies as low as 1 ms, a crucial metric for true real-time interactive and mission-critical applications. This is achieved through a more flexible OFDM waveform with scalable numerology and a clean-slate 5G Core (5GC) network. For high-bandwidth multimedia, 5G NR incorporates Massive Multiple-Input Multiple-Output (Massive MIMO) antenna arrays and Intelligent Beamforming,

far exceeding LTE's standard MIMO to achieve unprecedented spectral efficiency and aggregate throughput.

The most significant difference in capability stems from 5G NR's expansion into millimeter-wave (mmWave) frequencies (Frequency Range 2, FR2, spanning tens of GHz). These extremely high frequencies offer vast, contiguous blocks of spectrum (up to 1 GHz channel bandwidth) that are unavailable at lower frequencies, directly translating to multi-Gbps peak speeds and massive capacity for dense environments [31]. While mmWave signals suffer from higher propagation loss and shorter range than LTE's sub-6 GHz spectrum, this drawback is counteracted by Intelligent Beamforming, which focuses the signal energy into narrow, directed beams to maintain link robustness and efficiency, providing the necessary high-bandwidth foundation to power immersive multimedia experiences [32]. Thus, 5G NR's combined technological advancements allow it to realize the three key performance demands—low latency, high bandwidth, and reliability—that LTE could only partially satisfy.

The transition across wireless generations reflects a fundamental divergence in scheduling philosophy, each presenting distinct advantages and disadvantages for real-time multimedia streaming. Cellular technologies, specifically 4G LTE and 5G NR, employ a centralized, command-based scheduling model, where the base station (eNodeB/gNodeB) controls all resource allocation using Physical DL Control Channel (PDCCH) grants. An example of this scheduling scheme can be seen in **Figure 2.5.** LTE’s subframe limits its ability to achieve truly low latency, making it adequate but not optimal for real-time applications. In contrast, 5G NR utilizes scalable numerology and mini-slots with an agile PDCCH to virtually eliminate contention latency, enabling Ultra-Reliable Low-Latency Communications (URLLC). Conversely, Wi-Fi 6 (IEEE 802.11ax) operates on a contention-based, distributed model using Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), which introduces inherent variability and latency due to devices having to compete for channel access.6 While Wi-Fi 6 uses OFDMA and improved coordination to significantly boost local capacity and reduce contention overhead in dense environments, its core non-deterministic access mechanism makes it less reliable than 5G for guaranteeing the stringent quality of service required for jitter-sensitive, wide-area real-time multimedia streaming.

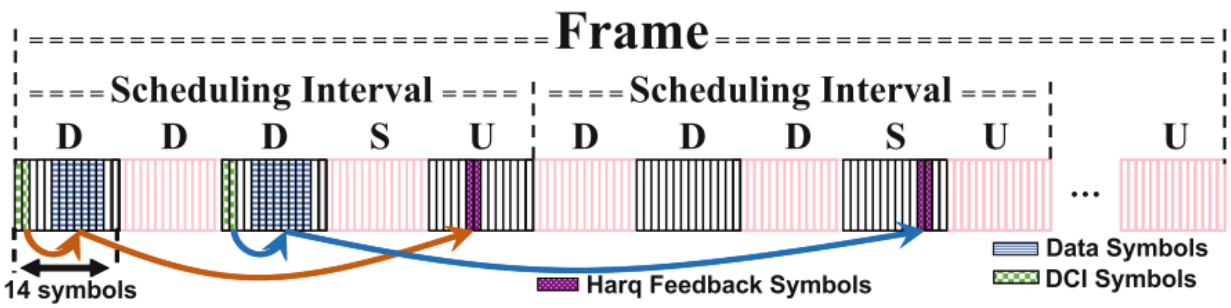

Figure 2.5: 5G NR mmWave PHY Scheduling Intervals: Adapted from work by Fezeu et.al.**,** this figure offers insight into how 5G NR approaches dividing its resources for multiple users on the PHY-layer [33]. Using a centralized scheduling system, where the gNodeB controls all data transmissions (via PDCCH and PUCCH signaling) and leverages scalable numerology to accommodate diverse service needs. The key to reducing latency is the flexible subcarrier spacing, which shortens the fundamental scheduling unit (slot) from 1 ms (like 4G) down to 0.125 ms for high-frequency millimeter-wave (mmWave) deployments. Also described in their work, are external factors that increase latency, including the asymmetric nature of UL/DL scheduling, the dynamic adjustment of data rates, re-transmissions due to poor channel conditions, and the power-saving techniques.

Performance Benchmarking Methods for Edge and Network Systems

To establish the background for this benchmarking framework's methods for measuring the performance of experimental edge systems, a discussion is given on the existing standards of objectives and metrics, established benchmarks, instrumentation, and measurement techniques. Furthermore, each edge application has requirements and constraints inherit to their environments and functions that need to be considered when designing a performance evaluation. The diagram in **Figure 2.6**, offers a classification of benchmarking frameworks organized by the aspects of the system that are analyzed. These aspects range from system design to real-world deployment factors.

The unified benchmarking framework proposed by this thesis adopts these dimensions to offer a comprehensive evaluation of application and network performance. By analyzing hardware and software capabilities of sensing nodes, network infrastructure, and edge resources, the framework addresses the system under test. The evaluation of user experience and protocol optimizations offer insight into the techniques analyzed. Conditions experienced at runtime, such as video resolution and physical environment, are also considered by the framework. By using these dimensions, the assessment can be designed with a clear direction.

Benchmarking Objectives and Metrics

Given the variability and complexity of these networks, rigorous performance

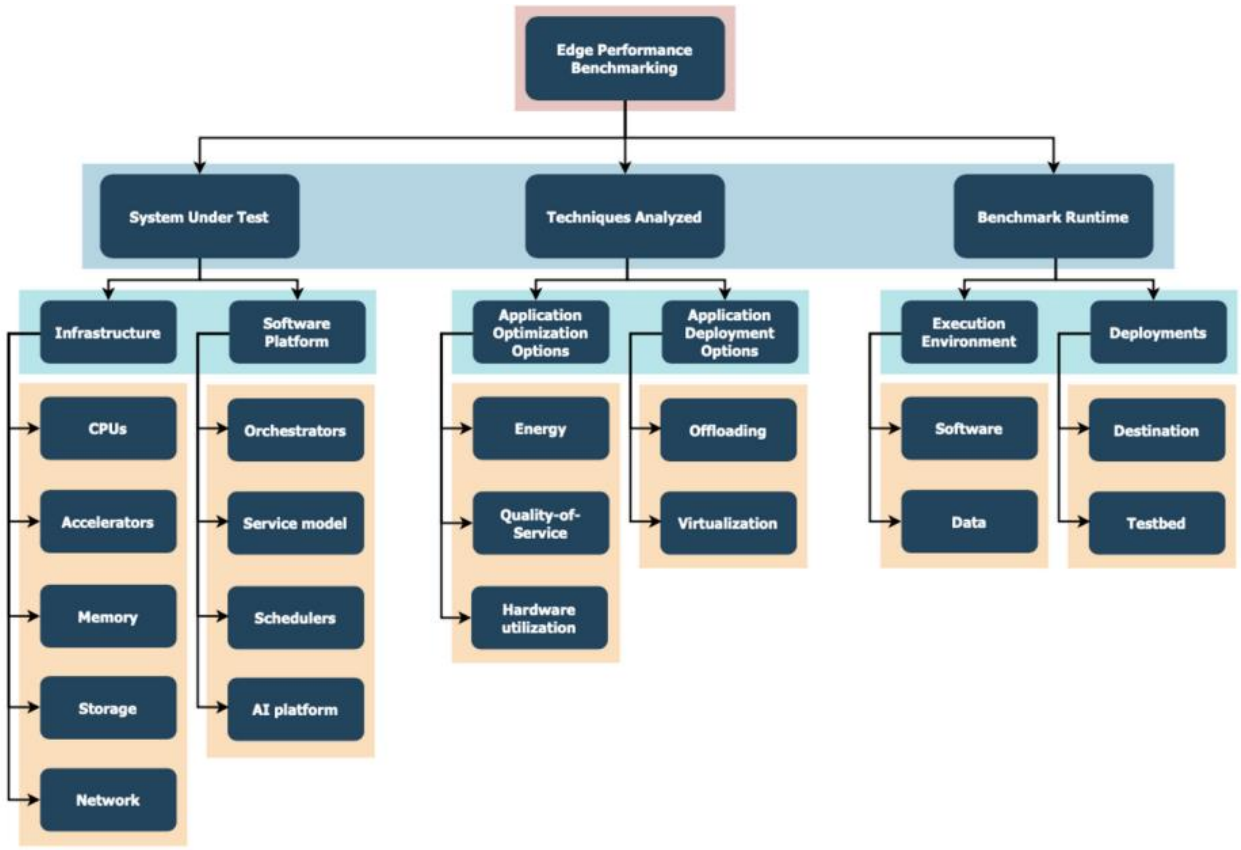

Figure 2.6: Edge Performance Benchmark Classification. Taken from [16], this diagram illustrates three primary dimensions used to evaluate edge computing configurations: system under test, techniques analyzed, and benchmark runtime. The system under test encompasses the physical and digital components of the platform, including processing, management, and communication resources. Techniques analyzed refer to the application-level and deployment-level optimizations that influence performance. Benchmark runtime captures the operational context, including execution environment and physical or cyber-physical conditions. Together, these dimensions establish a comprehensive framework for benchmarking edge resources and networks, integrating both system design and real-world deployment factors.

benchmarking frameworks are essential for quantifying system behavior under realistic conditions, providing reproducible and meaningful data for design and optimization. One standardized approach to objectively quantify the performance of wireless networks is described in the IEEE Standard for the Performance Assessment of Industrial Wireless Systems. They employ Performance Metrics (PM) as quantifiable measures used to objectively compare network performance across different technologies and configurations [34]. These PMs are broadly divided into two main categories: Test Characterization and Protected Network Performance metrics. The Test Characterization category, which ensures the accuracy and context of the measurement environment, is further structured into two sub-categories. The first

is Clock Synchronization Metrics, which ensures correlation between test events using measures like clock offset (difference from reference), clock skew (rate of offset change), and clock jitter (variation in offset), often relying on protocols like NTP (1 ms accuracy) or PTP (sub-microsecond accuracy). The second sub-category described in the IEEE standard, is Aggressor PMs, which track the progression of RF disturbances and are classified into path loss metrics, multipath metrics, and interference metrics. In other words, the Aggressor PMs are measurements that quantify the negative factors impacting a wireless signal during testing such as a weakened signal, defections and distortion, as well as the strength or type of noise or jamming.

The comprehensive framework described by the IEEE standard for performance assessment extends beyond characterizing the test environment to the actual performance of Industrial Wireless Networks (IWN) itself, which is categorized into data performance metrics and network management metrics [34]. The data performance metrics relate to the processing of data across various layers of the communication protocol stack, providing direct insight into communication quality. Key measurable data metrics include the Received Signal Strength Indicator (RSSI) and the Received Signal Quality Indicator (RSQI) (e.g. SNR or SINR), which address the physical layer integrity. Crucially, higher-layer metrics such as Packet Loss Rate (PLR), Round Trip Time (RTT), and Throughput quantify the network's efficiency and responsiveness to the application layer. The authors of the standard offer complementary PMs which are the network management metrics. These focus on the stability and operational protocols of IWNs and include the latency associated with establishing and maintaining connections, as well as overall network availability, ensuring the application is consistently responsive over the wireless medium.

Together, these metrics support the standardization of performance evaluation for wireless networks which are vital for the validity of the benchmarking framework proposed by this thesis. Through the dual angle approach of physical layer and network layer monitoring and analysis, a system can demonstrate its capabilities to support real-time multimedia communication.

Existing Edge and Network Benchmark Frameworks

This section critically reviews existing edge and network benchmarking frameworks, providing a foundation for the proposed unified edge performance benchmarking framework. A

variety of tools and methodologies have been developed to evaluate resource utilization, performance, and scalability in cloud and distributed systems, with notable examples including OpenStack, EdgeBench, and OpenRTiST [17, 35, 36]. These frameworks, while instrumental, typically focus on specific aspects—such as data center virtualization, as seen with OpenStack, or isolated edge performance metrics, as is the case with EdgeBench—often relying on simulated workloads or idealized network conditions. A detailed examination of their objectives, methods, and practical limitations reveals a critical gap: the absence of a holistic framework that integrates real-world data, realistic network topologies, and highly variable edge conditions to provide truly representative performance metrics. This thesis aims to address this deficiency by developing a unified framework that overcomes the shortcomings of the existing literature.

The OpenStack framework, while primarily a cloud operating system that controls large pools of compute, storage, and networking resources throughout a data center, has been extensively used as a platform for benchmarking cloud infrastructure, and by extension, the foundational layer for many edge deployments [35]. Its high-level design, seen in **Figure 2.7**, is a modular architecture comprising several interconnected services, which collectively enable the creation and management of virtual machines and containers. The objective of utilizing OpenStack in a benchmarking context is typically to measure the overhead and performance characteristics of its virtualization and orchestration layers, assess resource provisioning latency, and evaluate network I/O throughput within a controlled, centralized environment. The methods involve deploying standard benchmark suites along with the virtual infrastructure, allowing researchers to stress-test the underlying hardware and the hypervisor layer to establish performance baselines. However, its focus remains largely on the data center environment, making it less suitable for capturing the unique, highly distributed, and resource-constrained nature of the true edge.

In contrast, EdgeBench is a dedicated benchmarking framework specifically designed to evaluate the performance of computation offloading, resource management, and application execution in Mobile Edge Computing (MEC) environments [17]. Its high-level design provides a standardized, repeatable methodology for deploying and measuring various edge workloads, often incorporating both synthetic and realistic mobile applications across different edge infrastructures. The primary objective of EdgeBench is to provide quantifiable, comparable metrics—such as latency, energy consumption, and application-specific quality of service

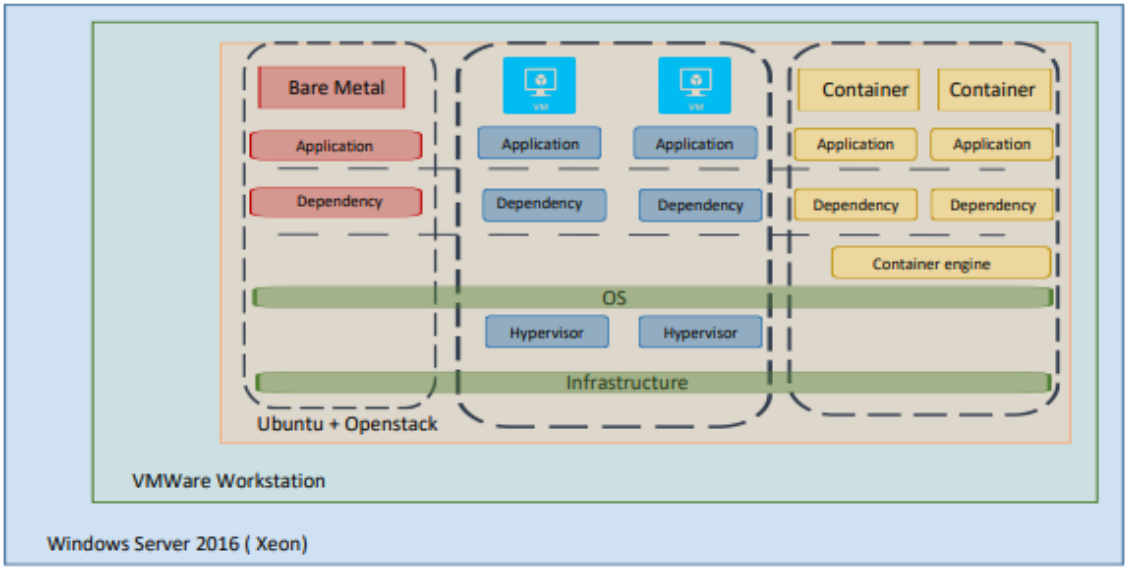

Figure 2.7: OpenStack Environment Architecture. Sourced from Dollinger et.al., OpenStack is a modular cloud operating system designed to deploy and manage large pools of compute, storage, and networking resources using decoupled APIs [35]. The architecture is organized around a Controller Node, which intercepts user requests, orchestrates the workflow across various services, and typically hosts the centralized dashboard. Compute Nodes manage the actual instances, supporting deployments like VMs, containers, and bare metal. This design separates component orchestration from device management, allowing for flexible and scalable infrastructure control.

(QoS)—that are directly relevant to edge computing scenarios. The methods employed involve a systematic approach to workload generation, infrastructure deployment, and data collection, often by emulating or utilizing real edge devices (like smartphones or IoT sensors) and network conditions. Crucially, while EdgeBench is a significant step toward realistic edge evaluation, its scope often remains limited to the performance of the offloaded application logic itself, and it typically uses simplified or static models for the highly dynamic and heterogeneous wide-area network that connects the edge to the core cloud.

Another related benchmarking framework that analyzes application and network performance is the OpenRTiST (Open Real-Time Style Transfer) framework, developed by George et.al. It is an open-source platform designed to evaluate the performance of real-time artificial intelligence (AI) inference in edge computing environments [36]. Its primary purpose is to provide a standardized and reliable method for comparing the performance of various deployment models, including device-only, edge-assisted, and cloud-based implementations, using metrics relevant to live and interactive applications. The framework employs a computationally demanding augmented reality application that applies artistic transformations to

live video streams through neural style transfer (NST). By utilizing the Gabriel platform to offload the intensive NST processing to an edge server, OpenRTiST enables the systematic measurement of key performance indicators—such as end-to-end latency and motion-to-photon delay—under a range of network conditions and hardware configurations.

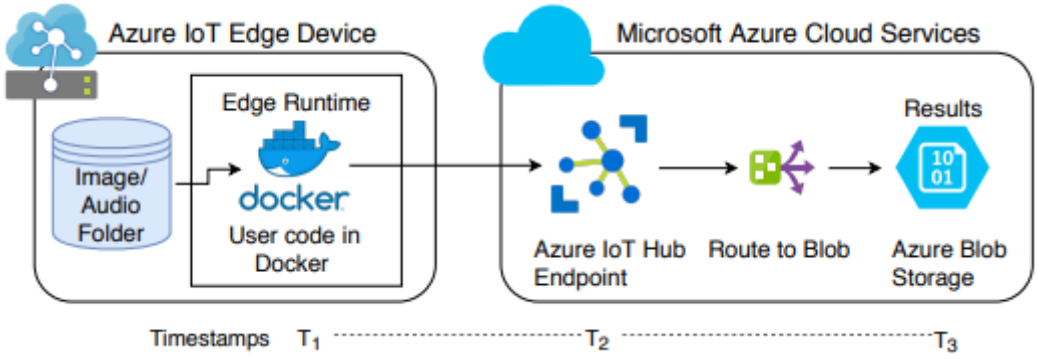

Figure 2.8: Example IoT Edge Architecture. This diagram was taken from the work done by Das et.al. [17]. EdgeBench is a specialized benchmark suite designed to evaluate the performance of applications in MEC environments using three canonical workloads: audio-to-text, image recognition, and a scalar sensor emulator. Its architecture involves edge devices sending data messages to a cloud-based IoT Hub (like AWS or Azure), which then routes the payload to storage. Metrics like compute time, time-in-flight, and end-to-end latency are captured using timestamps added at the edge device, the IoT Hub, and the final cloud storage to systematically measure performance across the entire pipeline. The framework is also adapted for cloud-only serverless pipelines to provide a baseline comparison for the edge-based execution

. The benchmarking frameworks described in this section have provided key insight into the objectives, architecture, and experimental methods for objectively evaluating edge and network systems from physical and cyber-physical perspectives. The key idea driving the development of these systems is the need for quantifiable, comparable metrics across diverse computing environments to accurately assess resource utilization and scalability. A review of these frameworks identifies the necessary components for effective benchmarking, which include: a defined workload or application, a method for deploying experiment configurations, a mechanism for handling input/output data, and a systematic approach to capture end-to-end performance metrics. However, to move beyond idealized scenarios and provide truly representative results, future benchmarking—the focus of this thesis—must explicitly incorporate the complexities of real-world data, realistic network topologies, and highly variable edge conditions.

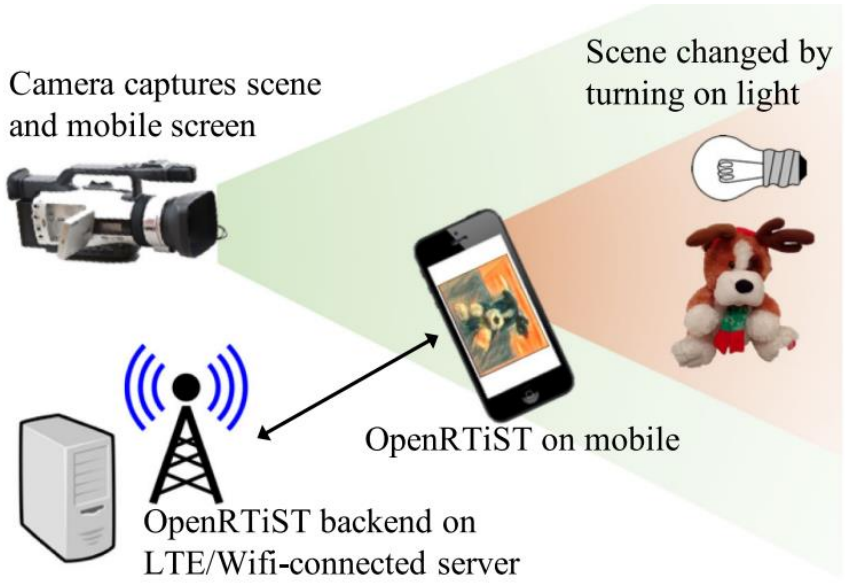

Figure 2.9: OpenRTiST Experimental Setup. Taken from the original paper by George et.al. [36]. The OpenRTiST architecture is a client-server design centered on the Gabriel framework for mobile cognitive assistance. A mobile client (the input) captures a live video feed and offloads frames to the edge server via the network. The server runs a Deep Neural Network (DNN) for real-time style transfer, which processes the frames and returns the transformed video (the output) to the client for display. This real-time interaction benchmarks end-to-end latency by measuring the time from video capture to the display of the processed, stylized frames.

Instrumentation and Measurement Techniques

Accurate and representative performance benchmarking of edge and network systems fundamentally relies on robust instrumentation and measurement techniques capable of capturing comprehensive metrics across multiple system layers. This section provides a detailed review of the methodologies employed to observe and quantify performance characteristics, addressing the three critical dimensions of a functional edge deployment. First, Application and End-to-End (E2E) Performance requires techniques such as packet capture, detailed event logging with precise time-stamping, and trace analysis to define latency and flow across the entire execution chain. Second, at the System Resource level, instrumentation must quantify fine-grained utilization metrics, including CPU, GPU, memory, and storage I/O, and energy consumption. Finally, Network Performance necessitates the measurement of dynamic communication attributes like throughput, latency, jitter, and packet loss, while also accounting for complex real-world factors such as signal quality, mobility effects, and wireless interference. An understanding

of these current techniques is essential for designing the unified framework's data collection and analytical core.

Real-World Relevance

Benchmarking wireless networks and edge infrastructure is critical for understanding and ensuring the performance, reliability, and responsiveness of mission-critical applications. As industries increasingly rely on real-time decision-making—ranging from autonomous vehicles and industrial automation to emergency response and healthcare monitoring—any degradation in network or system performance can have significant operational, safety, or financial consequences. Systematic measurement and evaluation allow engineers to quantify performance limits, identify bottlenecks, and optimize resource allocation, providing the confidence required to deploy high-stakes applications in dynamic, real-world environments.

Streaming Protocols and Compression Techniques for Wireless Edge Systems

Finally, the deployment of mission-critical services at the wireless network edge, as examined in this thesis, requires a careful equilibrium between reducing communication latency and optimizing bandwidth utilization. This section provides the technical groundwork for the key components analyzed in this study: the transport protocol and the video compression standard. Because the experimental framework employs UDP to isolate the raw network's impact on system efficiency, the supporting literature must substantiate this choice in comparison to more reliable alternatives. Additionally, the selection of H.264 and H.265/HEVC as the principal codecs for benchmarking necessitates a targeted evaluation of their respective trade-offs, particularly regarding compression efficiency and the computational demands imposed on resource-limited edge devices.

Real-Time Streaming Protocols

This subsection introduces the fundamental transport layer protocols—Transmission Control Protocol (TCP) and User Datagram Protocol (UDP)—and examines their implications for real-time communication at the network edge. The discussion emphasizes the inherent trade-off between TCP's reliability, achieved through guaranteed delivery and retransmission mechanisms that introduce variable and often substantial latency, and the stringent sub-second latency requirements of time-critical edge applications. The literature consistently identifies UDP as the foundational protocol for most real-time streaming systems (such as RTP/WebRTC), due to its connectionless, low-overhead design that prioritizes timeliness over reliability [37]. This

characteristic makes UDP the most appropriate choice for isolating network efficiency in the methodological framework of the present research.

Compression vs. Computation Trade-offs

The section then examines the key video compression standards employed in this study—H.264 (AVC) and H.265 (HEVC)—with a focus on their comparative performance and adoption within edge environments. H.264 is recognized for its broad compatibility and low computational complexity [38]. In contrast, H.265/HEVC offers markedly improved compression efficiency, typically reducing bitrate by 25–50% relative to H.264 for equivalent visual quality [38]. However, this gain comes at the cost of significantly greater computational and energy demands during both encoding and decoding, which can pose limitations for low-power, resource-constrained edge platforms.

Protocol and Codec Optimizations for Wireless Communications

Although this thesis employs raw UDP as an experimental baseline, the discussion further situates this choice within the broader context of contemporary systems that extend UDP's capabilities to achieve robust real-time performance. This includes an examination of protocols such as WebRTC and SRT (Secure Reliable Transport), which enhance UDP's lightweight transport model with application-layer reliability mechanisms, including sequence numbering, forward error correction (FEC), and adaptive jitter buffering [37]. Finally, the review briefly addresses the concept of Adaptive Bitrate (ABR) streaming to highlight that despite differing implementations, the principle of dynamically adjusting video quality remains essential for mitigating bandwidth variability in unstable or capacity-limited wireless edge environments.

Conclusion

This chapter established the technical and methodological foundations required to evaluate real-time wireless edge systems for mission-critical applications. Through review and synthesis of the existing literature, three core conclusions emerged. First, the migration of computation from centralized cloud infrastructure to the edge represents a deliberate architectural strategy to bound latency, reduce backhaul contention, and enhance reliability in time-sensitive domains. Second, the performance of these systems is fundamentally shaped by the characteristics of the wireless substrate itself. Wi-Fi 6, 4G LTE, and 5G NR each embody distinct scheduling philosophies and propagation behaviors that influence achievable throughput, stability, and end-to-end delay. Third, effective benchmarking of such environments demands

multi-layer, time-synchronized measurement conducted under realistic and occasionally adverse conditions. Even modest interference, congestion, or link variability can distort results derived from idealized setups, underscoring the necessity of integrated, cross-layer observability.

From these findings, the review distilled a set of practical design principles for the benchmarking framework developed in this thesis. The proposed methodology calls for strict synchronization between application and link-layer metrics, concurrent instrumentation of both compute and communication processes, and deliberate modeling of network and radio impairments. It further emphasizes experimental control through repeatable workloads, portable hardware, and factor-isolated trials spanning multiple radio access technologies. These perspectives address the methodological gaps identified in prior frameworks, which often neglect synchronized measurement or realistic wireless dynamics. The resulting benchmark advances the state of practice by coupling live application workloads with detailed link telemetry, thereby enabling causal attribution of performance degradations across the full stack.

Building on this foundation, Chapter 3 translates these principles into a unified real-time benchmarking system. The implementation integrates per-frame timestamping within RTP streams, synchronized logging of radio and application metrics, and controlled experiments across Wi-Fi 6, LTE, and 5G environments. This system embodies the conceptual framework outlined here and provides the empirical basis for the analyses presented in Chapter 5, where the hypotheses defined at the close of this chapter are systematically tested and evaluated.

CHAPTER THREE
BENCHMARKING SYSTEM DESIGN AND IMPLEMENTATION

The design of the benchmarking system utilizes the client and server paradigm where the server participates passively while the client initiates communication and data transfer. Both systems serve as metric data collection actors. Together, along with the underlying network, wireless interfaces, and physical environment, the components comprise the framework for a unified tool for discovering characteristics of system performance and relationships between applications and communication.

With a focus on modularity, reproducibility, and practical applicability, the benchmark in this thesis utilizes commercially available hardware and a custom-built streaming and analysis program. The hardware and software products used are well documented, portable, and intentionally selected to mirror systems deployable in industrial and field environments. Furthermore, the main features of the benchmarking system include precise measurement of multi-layer performance metrics, real-time monitoring, fault injection, and application parameter tuning. This framework provides a scalable and extensible foundation by decoupling application logic from instrumentation and maintaining flexibility across deployment contexts.

Overview of Benchmarking Framework

High-level Architecture

The high-level architecture, depicted in **Figure 3.1,** of the developed benchmarking framework is composed of three principal elements: a sensing node, a network black-box, and a system controller. The sensing node incorporates a single-board computer, responsible for interfacing with an image sensor via HDMI 2.0 and a wireless modem through USB 3.2. This node establishes communication with the system controller by leveraging the black-box network. The black-box network itself is architected from a radio interface, a gateway and router, and a wired interface, with interconnections facilitated by both coaxial cable and fiber optic connections. The system controller, manifested as a full desktop system coupled with a standard HDMI monitor, connects to this network via a wired medium. Beyond these tangible components, the framework further encompasses several non-physical elements, including a visualization and control graphical user interface (GUI), a supplemental network monitor application, and a physical impairment space that serves as the crucial link between the sensing node and the black-box network.

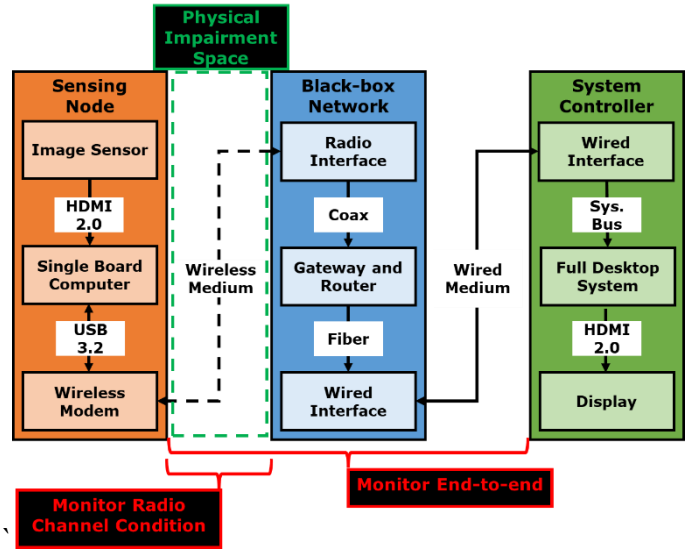

Figure 3.1: High-level System Design of the Benchmarking Framework. The architecture of this system utilizes three main components to benchmark system performance in a network edge scenario. The sensing node leverages an image sensor, SBC, and wireless modem to capture and transmit sensitive data. Depending on the capabilities and conditions of the intermediate network, the system controller will observe the incoming data stream at a measurable rate and delay. It is here that the user will visualize the live performance of the system.

This architecture allows for modular components that can be interchanged between experiments. Furthermore, the processing power can be modified to generate varying data rates. Modifying the network may also enable a larger and faster communication pipeline between the client and server. However, the system controller in this architecture serves as a more rigid component that would not significantly impact the benchmarked performance; rather, it functions as a control across experiments. Additionally, the framework attempts to isolate the device responsible for edge processing from the network that carries valuable data. This isolation is beneficial to the objective of studying relationships between the network and the critical service application. By understanding the wireless communication channel's condition and the end-to-end performance, the benchmarking framework provides insight into what can be achieved by a system's configuration and what would be expected in real-world environments.

The physical impairment space, lying between the Sensing Node’s antenna and the Black-Box Network’s radio front-end, serves as an RF sandbox for signal obstacles. Pairing these physical obstacles with software impairments yields a full factorial test matrix covering best-case to worst-case conditions.

In addition to the obvious components of this framework, there are services deployed to facilitate time-synchronization, systematically run scenarios, and manage the database of results. Network Time Protocol (NTP) client and server are deployed to keep the Sensing Node's and System Controller's system clocks synchronized within 500us to enable per-frame one-way latency to be captured without dedicated time-synchronization components. A master Bash script is used to configure experimental runs, and an offloading database tool is used for long-term storage.

Functional Breakdown of System Components

Figure 3.2 outlines the main functions of the system components. The sensing node is responsible for originating data traffic and capturing observed metrics. It directly interfaces the wireless communication channel to capture physical-layer metrics such as Reference Signal Received Power (RSRP), Signal-to-Noise Ratio (SNR), and transmission power. Simultaneously, it captures a continuous live video feed through an image sensor, and it quantifies key streaming performance indicators such as bitrate, latency, and loss. Additionally, the Sensing Node applies application-specific processing to the live video, manages the transmission of the video stream to the system controller, and logs all captured metrics for post-analysis.

Furthermore, the Black-Box Network is a controlled and abstract component which facilitates secure, managed, and routed communication between the client and server. Its internal structure remains opaque to the rest of the system during operation, but it includes a gateway, router, and modulation interfaces that shape the underlying network conditions. This layer enables the application of controlled impairments to emulate realistic or stress-induced network scenarios. Serving as the mediator between client-server interaction, this component is essential to the system design, but its internal mechanisms will not be explicitly exploited.

The System Controller ultimately stands as the terminal endpoint of the benchmarking pipeline. It receives the live video stream and renders it through the Graphical User Interface (GUI) for immediate visualization of content integrity and performance dynamics. Furthermore, the stream's effective quality is measured concurrently through received or end-to-end performance metrics such as bitrate, framerate, and jitter. The captured metrics are visually displayed and recorded. The System Controller serves as a central observatory for the experiment, enabling real-time monitoring and retrospective evaluation of both communication fidelity and application robustness.

Collectively, this division of responsibilities across the sensing node, network black-box, and system controller forms a tightly integrated framework capable of isolating and examining the influence of network characteristics on application performance under edge computing paradigms.

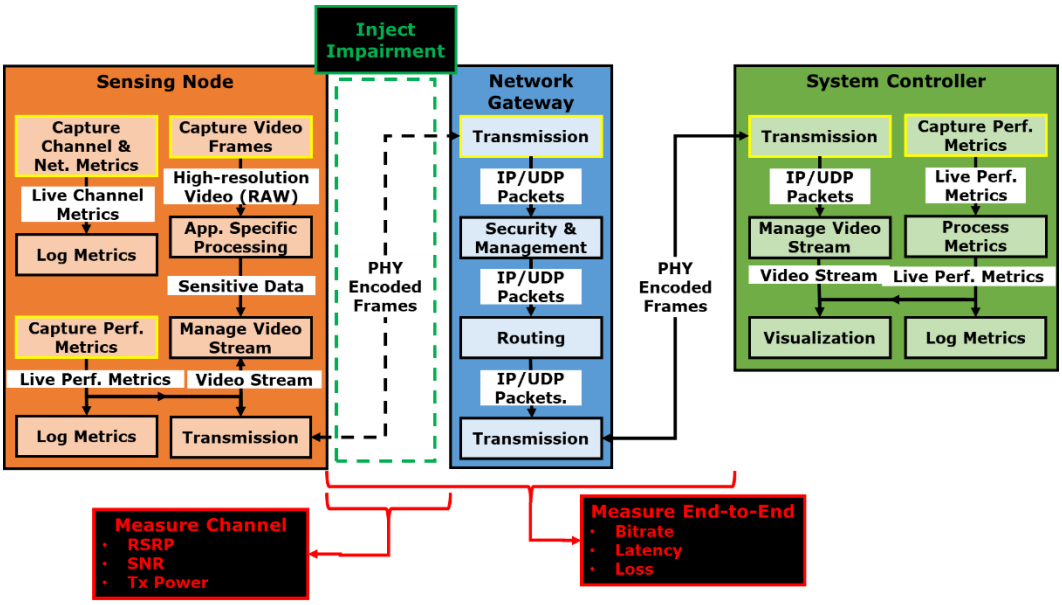

Figure 3.2: Functional Diagram of the Benchmarking Framework. The three main components of the benchmarking framework, namely the sensing node, network gateway, and system controller collectively participate in the experiment. The sensing node is responsible for video capture, metrics and performance monitoring, and streaming data. The network gateway facilities communication between the sensing node and system controller. The system controller receives streamed video, collects further metrics, and serves as the hub for visualization and control.

The functions of the sensing node and system controller are two-fold. Their primary responsibility is to act as a client and server, respectively, to exchange critical data across the network, simultaneously recording their performance using end-to-end metrics. Their secondary responsibility is to monitor the network during operation. This dual function allows for the direct comparison of critical application performance to prevailing network conditions. Moreover, while the application under testing will attempt to transmit data optimally, the experiment's design will counteract this by injecting impairments and forcing less efficient transmission methods.

Data Flow and Processing Pipeline

To clearly outline data origination, transformation, and its utility to the system, a comprehensive data flow and processing pipeline was designed and is illustrated across three

diagrams. **Figure 3.3** details the operations of the critical service application. Subsequently, **Figure 3.4** elaborates on the data flow of the network monitor tool. Finally, **Figure 3.5** combines these two pipelines, showcasing their collaborative functionality. Each of these diagrams will be elaborated upon individually, beginning with the critical service application.

During the data capture stage, live video is acquired from an optical sensor utilizing the Video4Linux2 (V4L2) interface in conjunction with the GStreamer multimedia framework. Video samples then traverse the GStreamer pipeline to an application-level sink, where the lightweight MobileNetV2 model performs facial feature extraction using a single-stage detector known as RetinaFace. This process marks the transformation of raw video data into semantically meaningful information relevant to the system's intended application.

Following feature extraction, the system applies a high-precision transmission timestamp to each processed frame. These frames, now embedded with metadata, are transmitted across a wireless interface, which may vary based on the experimental setup and could include 5G, LTE, or Wi-Fi links. During this transmission, the sensing node concurrently monitors the streaming process, collecting the egress bitrate performance and logging for future analysis

After propagating through the network, the data is received through a complementary GStreamer pipeline. A second timestamp is assigned to each frame upon reception, allowing the system to compute end-to-end latency by comparing it to the corresponding transmission timestamp. The video, now fully annotated and performance-aware, is routed to the GUI where it is rendered for visualization. Meanwhile, the performance data is processed and continuously updated on a real-time plot, offering insight into the behavior of the stream.

Running in parallel with this critical application is the data pipeline associated with the network monitoring subsystem. This component observes the communication environment by sampling telemetry related to both the radio link and the network core. Parameters such as received signal strength, interference levels, channel quality, and transmission power are captured by the sensing node and sent to the system controller. There, they are synchronized with the application metrics and displayed through the GUI. This simultaneous presentation allows observers to correlate the behavior of the application with the underlying condition of the network.

The third and final diagram provides a synthesized view of both pipelines operating in unison. The critical service application pipeline emphasizes the real-time performance and

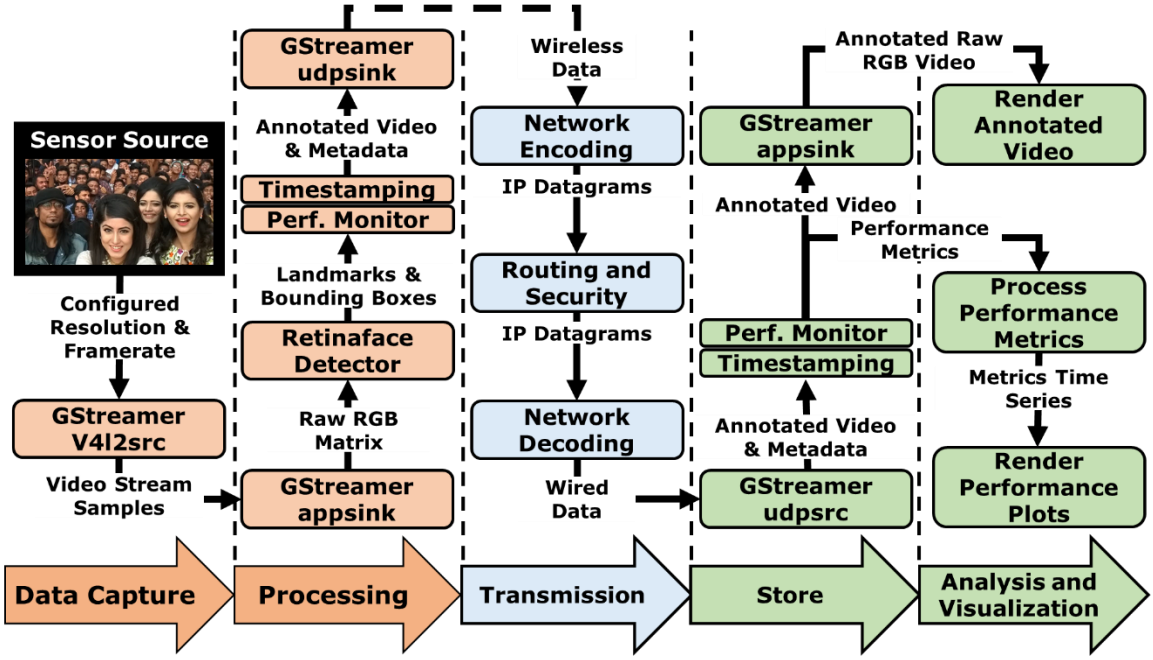

Figure 3.3: Data Flow and Processing Pipeline of the Critical Service Application. Data of the Critical Service Application flows through five main stages: data capture, processing, transmission, store, and analysis and visualization. The orange segments represent the sensing node, blue represents the network, and green represents the system controller.

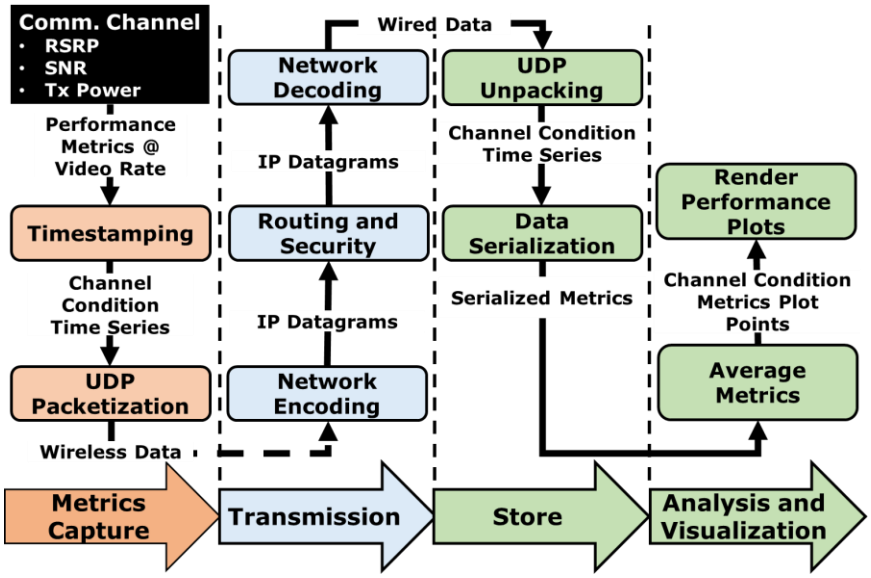

Figure 3.4: Data Flow and Processing Pipeline of the Network Monitor Tool. Data of the Network Monitor Tool flows through four main stages: metrics capture, transmission, store, and analysis and visualization. The orange segments represent the sensing node, blue represents the network, and green represents the system controller.

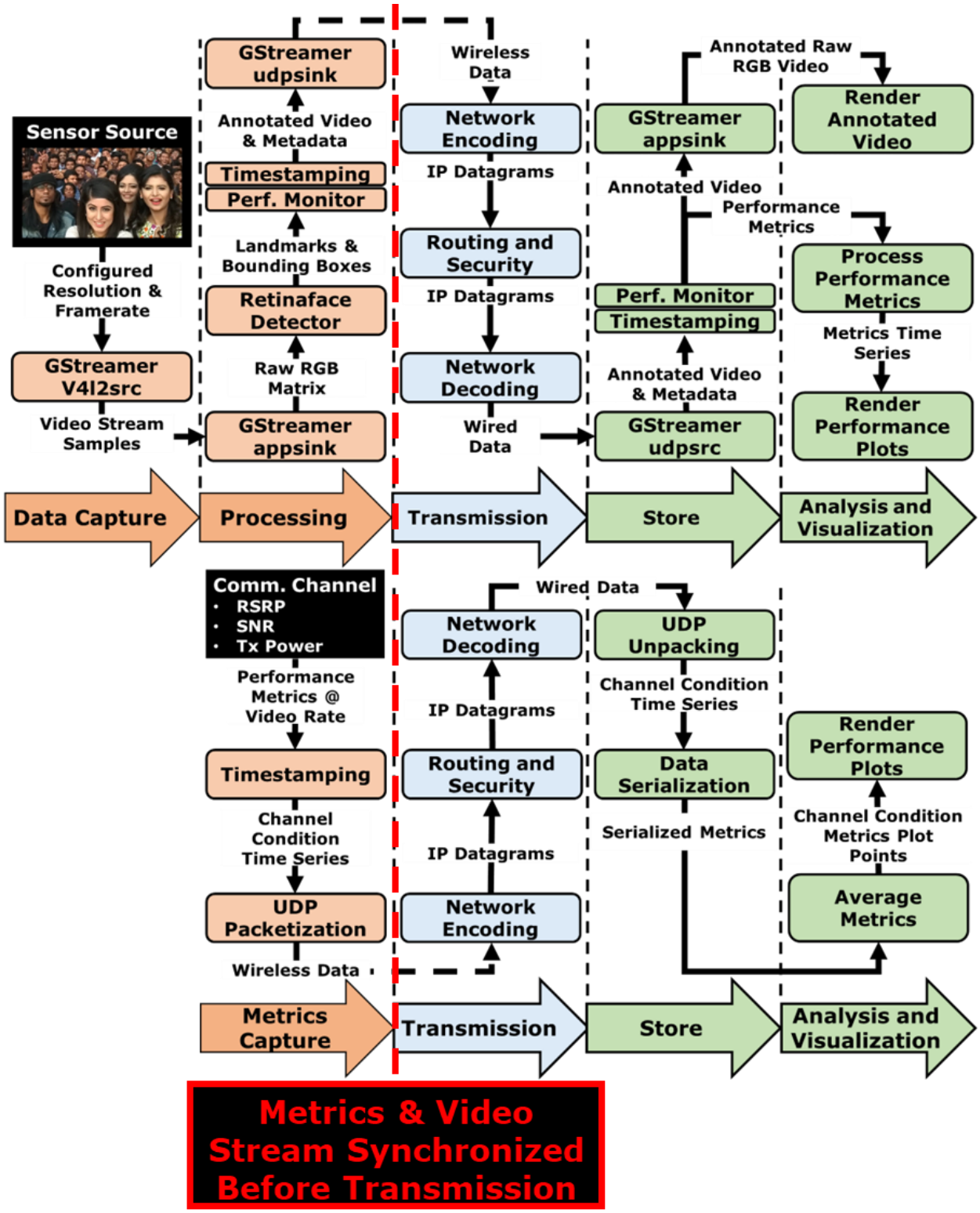

Figure 3.5: Combined Data Flow and Processing Pipeline of the Benchmark Framework. This highlights the interoperation between the critical service and the network monitor tool. Data captured from each are synchronized before transmitting over the network allowing the system controller to display both in real-time.

reliability of an edge-deployed application, while the network monitoring pipeline contextualizes that performance with a live characterization of the communication channel. Together, these interwoven processes allow the benchmarking framework to explore and quantify how network conditions influence application behavior at the edge.

Through this integrated architecture, the system can reveal subtle and important trends in end-to-end performance. By tightly synchronizing application metrics with network telemetry, the framework enables empirical studies of edge systems under realistic, variable, and sometimes degraded communication environments. This design ensures that each experimental run yields meaningful data about the limits and capabilities of low-latency, network-dependent computing at the wireless edge.

By synchronizing the data pipelines prior to transmission, the framework ensures that both types of data are temporally aligned upon arrival at the system controller. This enables cohesive, real-time rendering of performance metrics alongside streaming video, offering a unified view into the interplay between network conditions and application behavior. Such integration is critical for accurately assessing the resilience and responsiveness of edge computing applications in dynamic, heterogeneous wireless environments. The figure reinforces the framework's capability to generate holistic performance insights by bridging critical service functionality with real-time network observation.

Critical Service and Network Monitoring System and Application

In this section, we outline the technical details behind the design and implementation of the critical service and network monitoring system and application. The former operates as a real-time, performance-sensitive edge workload, and the latter operates as a passive capture and analysis tool to the condition of the underlying network in which the workload is running. Together these applications form the functional backbone of the benchmarking system. seen in **Figure 3.6**, providing both the subject of the experiment and the means to monitor itself.

The critical service application, operating as client on the sensing node, and server on the system controller, is designed to simulate a mission-critical edge AI task. It will perform on board real-time facial feature detection and streaming. It captures a live video feed, performs lightweight ML inference, and transmits the annotated video stream to a remote system controller. The data stream is processed and transmitted in real-time, also key performance

indicators (KPIs) such as frame rate, transmission latency, jitter, and bitrate. These KPIs are collected at the source (sensing node) and destination (system controller) and provide direct insight into the application's responsiveness and quality of service.

In parallel, the network monitoring application continuously observes the characteristics of the underlying wireless communication link. It collects low-level metrics including Reference Signal Received Power (RSRP), Signal-to-Noise Ratio (SNR), and transmitted power, along with network impairments such as packet loss and retransmissions. These metrics provide a granular view of the channel and link-layer behavior, helping to contextualize performance degradations in the critical service stream.

These two applications are married at the system controller, where the metrics collected from both the critical service performance and the communication channel condition, are time-aligned, visualized, and stored for further analysis. This tight integration allows the benchmarking framework to uncover correlations between environmental and network conditions and the behavior of delay-sensitive applications. For instance, a spike in jitter or a drop in RSRP can be directly tied to a drop in video quality or an increase in end-to-end latency.

Ultimately, by simultaneously orchestrating an active edge workload and a passive network monitor, the system creates a comprehensive testbed for evaluating the co-performance of intelligent applications and the networks they rely upon, especially under real-world constraints and dynamic conditions.

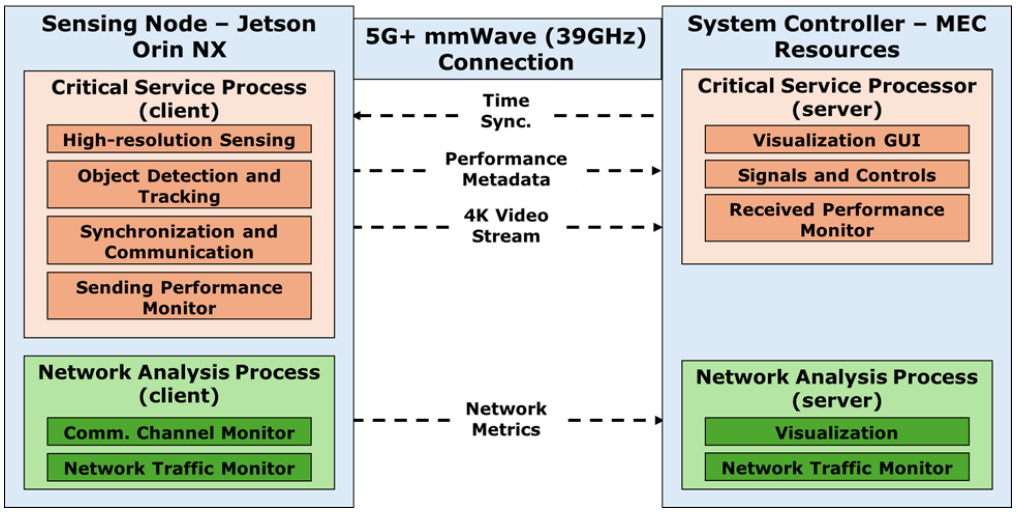

Figure 3.6: Critical Service and Network Monitor Processes. This figure illustrates the main functions of the client and server processes for both the critical service and network monitor applications. For each application, the client executes on the sensing node while the server side executes on the system controller.

Hardware (Specs, Interfaces, Rational)

The hardware components of the benchmarking system include processing platforms, video sources, and wireless modems. Each physical component deployed in the experiments of this thesis were chosen for their distinct traits like the size, weight, power, and cost (SWAP-C) characteristics shown in **Table 3.1**. We highlight the relevant specifications, interfaces, and design considerations for the devices used.

Sensing Node Processing Platform

Resources are often tightly constrained at the network's edge, therefore the processing platforms used for the sensing node, like those shown in **Figure 3.7**, are small, lightweight, and low-power systems. Specifically, three systems were chosen, the Radxa Rock 5B and Jetson Orin NX, and a more powerful Mini-PC option, the Intel Core i7-1360P NUC Mini-PC.

System Controller Processing Platform

Used as a control in the experimentation of this thesis, the system controller is embodied as a Dell Precision Desktop PC, shown in **Figure 3.8**. For the experiments conducted in this research, the system controller does not require extensive processing capabilities.

Video source

Visual input to the sensing node is provided by a continuous HDMI 2.0 video source. Its resolution can be varied from 360p to 2160p with framerates of 21fps to 120fps to offer the system under testing a wide range of workload sizes. HDMI is an ideal choice for this design due to its extremely high data rates and reliability. The video source also enables quick reconfiguration of the video content and complexity.

Table 3.1: Size Weight Power and Cost (SWaP-C) Characteristics of Computing Platforms.

Platform	Size (L x W x H)	Weight (grams)	Power	Cost (USD)
Radxa Rock 5B	100mm x 75mm x	200	20W	~200
Jetson Orin NX (Module Only)	69.6 mm x 45 mm	28	25W	~600
Intel NUC Mini-PC	117mm x 112mm x 54 mm	502	120W	~600



Figure 3.7: Processing Platforms of the Sensing Nodes. The devices shown in this figure are the processing platforms integrated into the sensing nodes. These platforms perform video capture, data stream management, and application specific processing. The top image depicts the Jetson Orin NX edge AI computer. On the bottom left is the Intel NUC i7-1360p mini computer. The Radxa Rock 5B RK3855 single-board-computer is shown in the bottom right image.

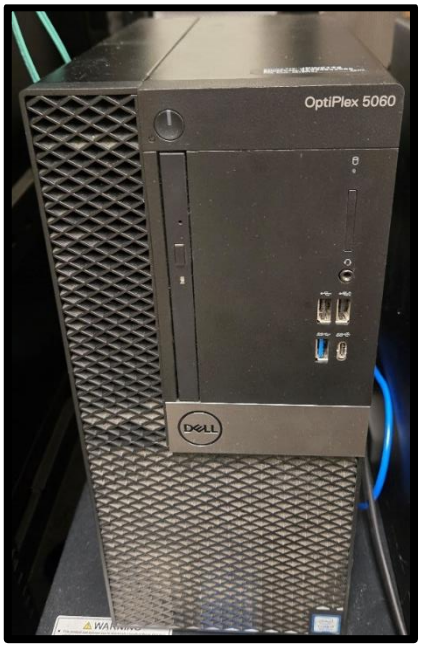

Figure 3.8:Processing Platform of the System Controller. The device depicted in this image is used as the processing platform for the system controller. The Dell OptiPlex 5060 is a general-purpose desktop system equipped with an Intel Core i7-8700 and 16GB DDR4 RAM. The system controller will display information and interact with users through its display connected via HDMI 2.0.

Wireless Modem

Three wireless modems were selected to fulfill the role of a wireless network interface. Each of the modems interface with the sensing node via a USB 3.2 connection. Firstly, the Wi-Fi 6 TP-Link nano supports **X** speeds. Secondly, the Quectel RM530N-GL 5G modem is used for both the 4G LTE and 5G mmWave scenarios. There are two different carrier boards, one which enables 5G mmWave connectivity and the other which does not.

Video Streaming

A vital aspect of the critical service application is its ability to transmit high-bitrate multimedia data with a strong dependance on QOS guarantees. The sensing node's and system controller's streaming abilities were carefully designed and implemented to ensure these guarantees are maintained. To offer the benchmarking system a greater degree of flexibility, the data streaming methods were designed to expose key parameters that directly influenced performance and workload size including the resolution, framerate, streaming protocols, and compression techniques. The two core software frameworks used to capture live video and manage the streaming pipeline are Video4Linux2 (V4L2) and GStreamer. This section will highlight the design considerations and implementation details of the video streaming function.

GStreamer

The GStreamer multimedia framework provides a powerful and flexible system for constructing complex data pipelines, particularly crucial for network streaming and real-time multimedia processing within the critical service application. This software component acts as a high-level abstraction layer that simplifies the management of various multimedia tasks, ranging from capturing and encoding to decoding and rendering. By organizing these operations into a connected series of "elements," GStreamer enables seamless flow and transformation of data.

GStreamer's core strength lies in its standardized data pipeline model, which offers a modular and extensible approach to multimedia processing. It abstracts away the complexities of underlying hardware and low-level APIs like V4L2, allowing developers to focus on the logical flow of multimedia data. For a benchmarking framework, this abstraction is vital as it allows for rapid prototyping and modification of complex streaming scenarios without extensive low-level programming, ensuring the focus remains on system performance analysis.

Selecting GStreamer for real-time network streaming is an ideal choice for multiple reasons. Firstly, its plugin ecosystem provides a library of pre-built elements for virtually any

multimedia task, significantly reducing development time and effort. Secondly, its cross-platform compatibility and robust design contribute to the framework's overall flexibility and potential for future expansion. Lastly, its ability to seamlessly integrate with low-level capture APIs, such as V4L2, allows the framework to leverage both granular hardware control and high-level, efficient multimedia stream management, which is essential for comprehensive benchmarking.

The video processing pipeline designed for the benchmarking system is outlined in **Figure 3.9**, and it includes the video capturing and encoding pipeline of the sensing node client and the decoding pipeline of the system controller or server. GStreamer manages each pipeline element along with the synchronization and communication between elements.

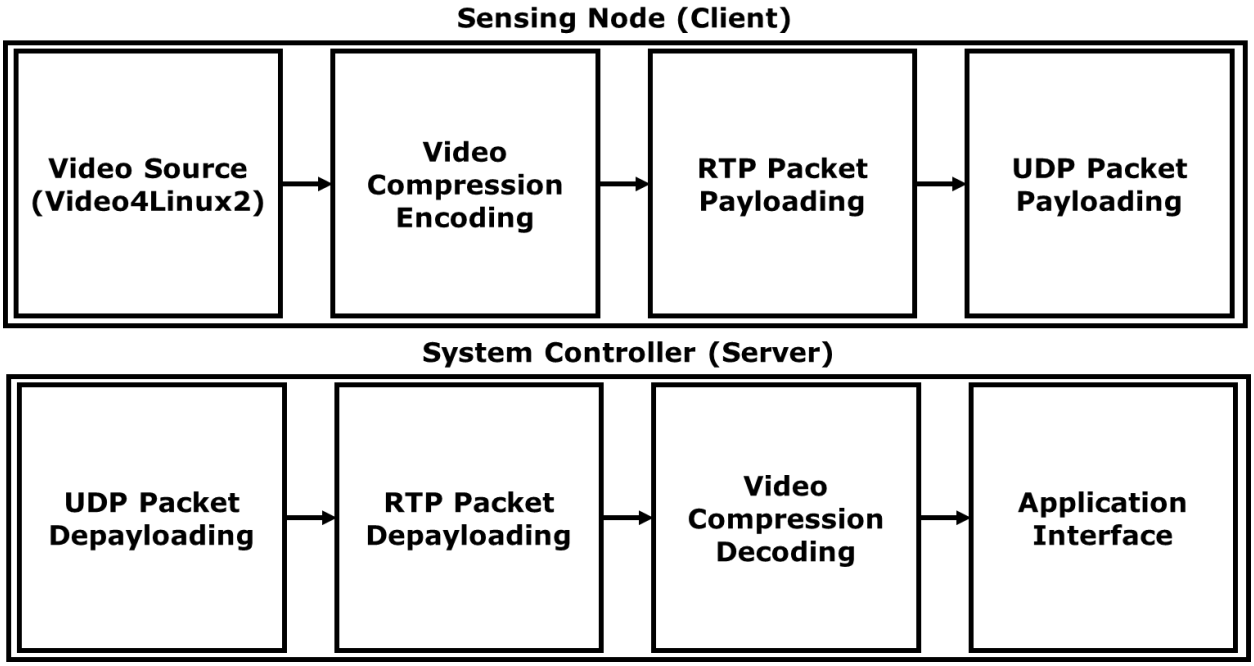

Figure 3.9: GStreamer Video Processing Pipelines for Client and Server. This figure gives more detail into what the GStreamer pipelines consist of within the sensing node and system controller. The client pipeline captures, encodes, and packetizes video frames into an RTP/UDP stream for transmission. Conversely, the server pipeline receives these packets, depacketizes the payloads, decodes the data back into video frames, and provides the reconstructed stream to the main application for display and analysis.

Live raw video frames are captured from the video source using the Video4Linux2 capture element. This element allows for control over framerate, resolution, pixel format, and other contributing factors. Data then flows to the encoding element where frames are correlated, and the specified algorithm compresses the stream. Video packets are assembled at the RTP payload element and further packetized using the UDP sink element. Packets travel over the

network to the system controllers UDP source element where the UDP payload is extracted. Video frames are reassembled from RTP packets and uncompressed by the decoding element. Raw video frames are read into the main programs memory where analysis and display are performed.

Not shown in **Figure 3.9** are inter-element queues or buffers which help absorb inconsistencies in processing time. Additionally, these queues offer a means for developers to apply custom processing. This is the location where the application specific processing is applied on the sensing node. GStreamer buffers are read into the main program, the raw video frame is extracted, and the pixels can be manipulated and injected back into the pipeline. GStreamer utilizes element pads to accomplish this. At the receiving and sending ends of an element are pads which connect to adjacent elements. This API exposes a non-synchronized methods to access these pads and monitor data traveling across them. The graphic in **Figure 3.10** demonstrates this mechanism.

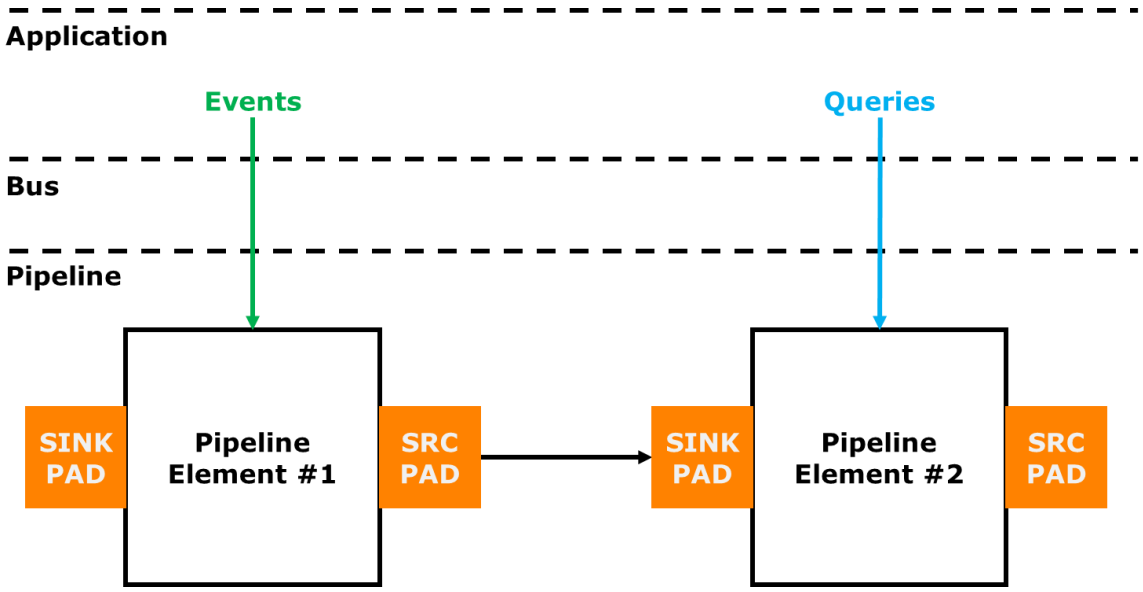

Figure 3.10: GStreamer Pipeline Element Pad Probes. This figure highlights an indispensable mechanism that allows a parent application to probe aspects of the pipeline elements, such as buffered frames in memory and readable writable properties. Most importantly, for the streaming application used in this thesis, this allows for direct access to video frames enabling bounding boxes to be overlayed. Additionally, it allows for the modification of network packets.

GStreamer pipelines operate as sub-processes of a parent process and expose methods to enable communication between the two. In addition to sending status messages to the application and messages between elements, it can receive events and queries initiated by the parent

application. It is this method that allows us to apply facial feature extraction to the frames that travel through the pipeline.

This mechanism is also leveraged to access RTP packets before they are packetized for UDP transport. **Figure 3.11**, offers a visual to explain how custom timestamps are embedded into the packet header of RTP packets.

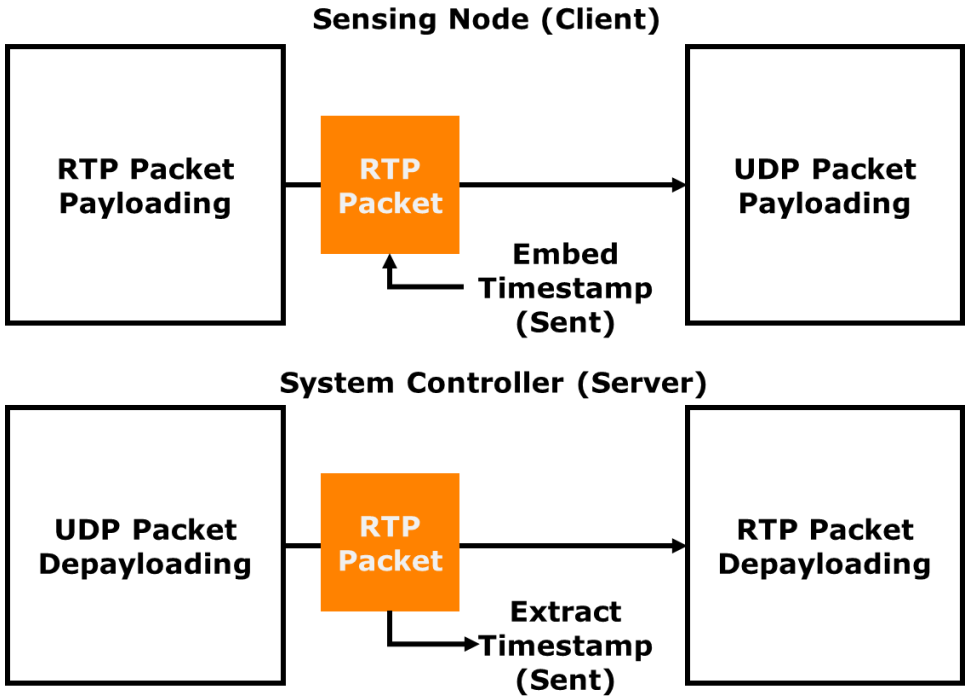

Figure 3.11: Embedding and Extracting Timestamps from RTP Packets. Building on **Figure 3.10**, this diagram shows at which elements RTP packets are accessed to embed and extract timestamps. These timestamps provide a temporal marker representing the moment in time that packets are sent for transmission. This method allows for the receiving device to calculate end-to-end delay.

As the sensing node constructs RTP packet it concurrently queries this data to expose the RTP header. The parent application clocks this moment and alters the header to embed the timestamp before it is sent over the network offering the system controller a reference point to calculate the end-to-end delay.

Video4Linux2

The system level interface to the kernel's video device file is managed by the V4L2 drivers and API. This software components allows the critical service application to have low-level control over the video source. Using this API, the application can easily reconfigure the resolution, framerate, pixel format, and buffer management.

The standardized video device interface V4L2 offers bypasses higher-level multimedia frameworks, offering direct access to raw sensor data streams. For a benchmarking framework, this kernel-level access is crucial as it minimizes software overhead and ensures that performance measurements reflect the true capabilities of the hardware and network, rather than being influenced by intermediate abstraction layers.

Selecting V4L2 for real-time video acquisition is an ideal choice for multiple reasons. Firstly, it is widely available on devices running the Linux Kernel, further strengthening the benchmarking framework's flexibility. Secondly, it is highly reliable and fault tolerant which strengthens the reproducibility between experiments. Lastly, it exposes low-level controls to the user that enables a wide range of input types and conditions.

Streaming Protocols

The transmission of multimedia data is heavily influenced by the network protocols used to manage video playback, synchronization, and reliability. For applications under tight time and quality constraints, this software component is crucial. The protocols used will dictate how frames are packetized, transmitted, and reassembled.

One protocol used in the implementation of this thesis is the Real-time Transport Protocol (RTP). The packet structure for this protocol is shown in **Figure 3.12**. This protocol is built from the User Datagram Protocol (UDP) and was designed for transmitting multimedia data across a network in real-time [39]. It does not offer any guarantees for timely delivery, quality-of-service, or out-of-order delivery, and it makes no assumptions about the condition of the network. Therefore, it is a barebones method to transmit a video stream across a communication link. This is an idea baseline for the experiments in this thesis because it leaves the focus of efficient transmission on the network.

The packet header offers a configuration bit to expose some customization. This bit is referred to as the header extension bit or 'X' and is located at bit 4. While the parent application accesses the RTP payload element in the GStreamer pipeline, it sets this bit to include custom information. These details are elaborated on in **Figure 3.13**.

By enabling the header extension bit, a variable header extension payload can be inserted into the RTP packet. For this application, the profile is left unmodified, the length is set to one to indicate a payload size of one 32-bit word, and the timestamp at which the data arrives to this point is copied into the header extension payload.

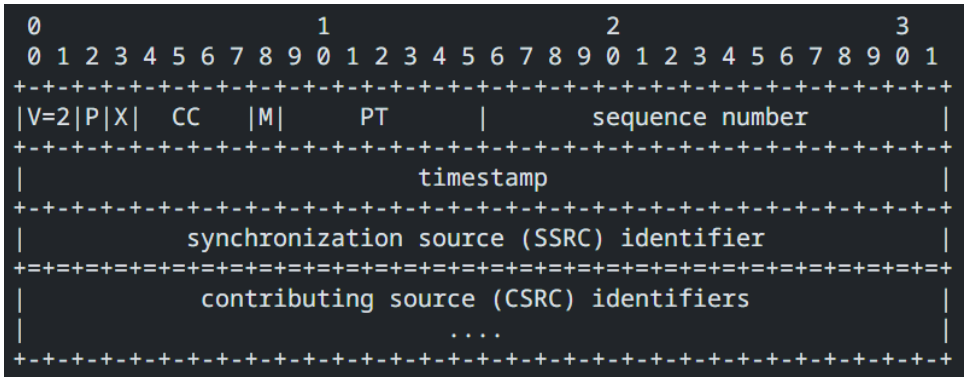

Figure 3.12: RTP Packet Fixed Header Fields. Taken from [39], this figure details the standard RTP packet header, which is essential for structuring and organizing the transmission of video frames over a network. The header contains a crucial customizable field, the RTP header extension, located at bit 4, which allows for additional information to be included in the packet.

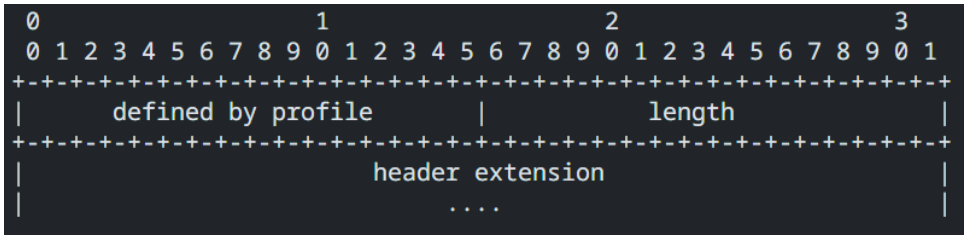

Figure 3.13: RTP Packet Header Extension. Taken from [39], this figure provides further detail on the RTP header extension mechanism. For this custom-built video streaming application, the extension is used to embed a timestamp that indicates when a packet is sent for network transmission. This timestamp enables the system controller to calculate the end-to-end delay. The length field is set to a value of one, which specifies a single 32-bit word in the extension payload.

Compression Techniques

Video stream compression (e.g., H.264, H.265, VP9) employs sophisticated algorithms to reduce the size of raw video data by removing redundant or perceptually less significant information. This process is crucial for minimizing bandwidth consumption and storage requirements, making high-quality multimedia feasible over constrained network links.

The integration of specific compression methods is paramount for the benchmarking framework for several reasons. Firstly, their selection directly affects the "bitrate, latency, and loss" metrics measured in the experiments, allowing for a precise understanding of their influence on application performance. Secondly, by subtly modifying and evaluating different combinations of these techniques, the framework can push the capabilities of the tested devices further, revealing the true performance achievable under optimized conditions. This detailed analysis of various protocols and compression levels significantly increases the framework's

applicability to diverse real-world scenarios and enables a more comprehensive understanding of system behavior.

Application-Specific Processing

Processing tasks conducted by edge computing system can vary widely from object detection to real-time analytics, however, one common goal is to move processing closer to the point at which data is collected. Because edge computing is combined with critical services that require high communication quality and reliability, processing tasks must be efficient and not introduce excessive delays. The example application is facial feature extraction, a task that involves identifying and localizing key facial landmarks within a captured video stream. Its implementation leverages the RetinaFace model, a robust single-stage detector renowned for its high accuracy in detecting faces and their corresponding landmarks, even under challenging conditions.

MobileNetV2 is a convolutional neural network architecture specifically designed for mobile and embedded vision applications. Its architecture is particularly relevant to edge applications due to its small memory footprint and minimal computational requirements. This design makes it highly suitable as a backbone for RetinaFace on resource-constrained edge devices, providing a balance between detection performance and computational load.

The inclusion of this application-specific processing is fundamental to the benchmarking framework for several reasons. Firstly, it represents a sensitive edge application that is directly impacted by changes in video quality and processing latency. Secondly, by executing this computationally intensive task on the sensing node, the framework can accurately quantify the end-to-end performance under varying network and processing loads. The performance of this specific deep learning inference task directly contributes to the overall "bitrate, latency, and loss" metrics, providing concrete evidence of the system's practical capabilities and the effects of dynamic environmental conditions on real-world edge AI applications.

To aid in the integration of RetinaFace and the critical service application, the open-source library NCNN was used. This library offers an easy-to-use API and a host of example implementations for several ML models. Its methods were specifically developed and optimized to tailor toward resource constrained mobile platforms. Furthermore, it enables the benchmarking application to readily except other ML computation tasks such as object detection, segmentation, and pose estimation.

System Controller Graphical User Interface

The GUI, displayed by the system controller, offers the user of the benchmarking system a source of control and visualization. From here, they can view the live video feed, monitor the key performance metrics, and terminate the experiment. The layout of this GUI is shown in **Figure 3.14**.

Performance Monitoring and Data Logging

This section details the methodologies employed for real-time collection and logging of performance metrics from video streaming and communication channel conditions. This capability is central to the benchmarking framework, enabling comprehensive analysis of processing platform capabilities, wireless communication technologies, and the intricate relationships between critical services and their underlying networks, especially under impairment.

Video streaming performance metrics are gathered by the critical service application at two pivotal points: the client (sensing node) and the server (system controller). These metrics are

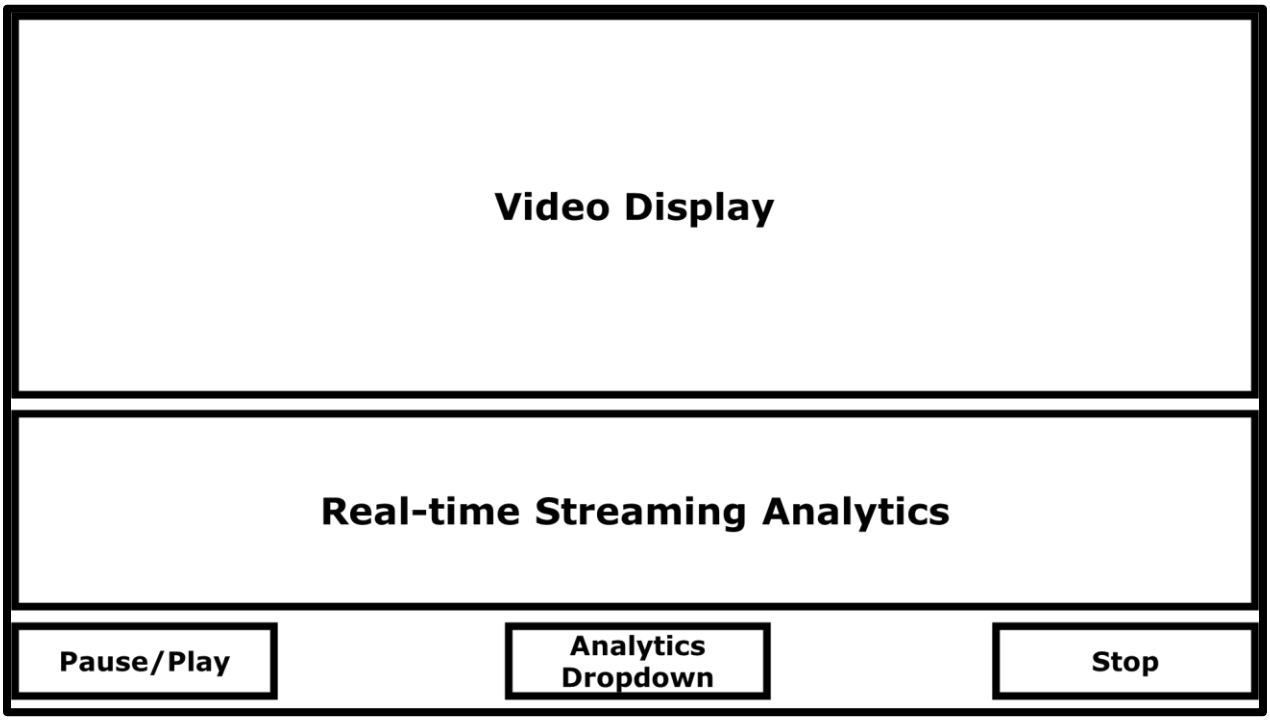

Figure 3.14: System Controller Graphical User Interface. Key components of the graphical user interface (GUI). The primary panel renders the incoming video stream. In the middle, a real-time analytics panel displays live plots of performance metrics. The control bar at the bottom contains buttons to play, pause, and stop the video stream, as well as to select the metric being visualized.

essential for a comprehensive understanding of the real-time quality of experience. To quantify end-to-end one-way delay, a high-resolution timestamp is embedded into the RTP packet header when video frames, within the GStreamer pipeline on the client, reach the "rtppay" element, marking the precise moment of data transmission. Upon arrival at the "rtpdepay" element on the server, another timestamp is recorded, signifying the exact time of packet reception. The difference between these two timestamps directly quantifies the end-to-end one-way delay, providing a direct measure of network latency affecting the video stream. Concurrently, the data size of each RTP packet is recorded at both the "rtppay" and "rtpdepay" elements, synchronized with the timestamps. These synchronized data points form a time series that enables the precise calculation of instantaneous sending and receiving bitrates of the video stream, offering a dynamic representation of the critical service application's performance. Furthermore, on the server, this collected time series is enriched with additional Quality of Service (QoS) metrics, specifically framerate and jitter. Framerate indicates the number of frames processed per second, providing insight into the smoothness of video playback, while jitter quantifies the variation in packet arrival times, directly reflecting network instability that can lead to perceived stuttering or freezing. The complete time series, encompassing these detailed metrics, is then displayed in real-time on a Graphical User Interface (GUI) and logged to a file for subsequent post-analysis.

Similarly, the network analysis application is responsible for continuously monitoring the condition of the communication link or channel. This is achieved by continually polling relevant metrics from the wireless modem via its serial interface. Key metrics such as Reference Signal Received Power (RSRP), Signal-to-Noise Ratio (SNR), and Transmit (TX) Power are polled at the same rate as the video frame capture. RSRP provides an indication of the strength of the received signal from the base station, directly affecting the quality of the wireless connection. SNR, on the other hand, measures the ratio of the desired signal power to the background noise power, which is a critical determinant of data transmission reliability and speed. TX Power represents the power at which the client's wireless modem is transmitting, offering insights into the device's power consumption and potential for interference. This synchronized polling rate with video frame capture allows for a high-resolution analysis of the dynamic state of the communication channel. This data is meticulously timestamped with high precision, forming a dedicated time series that enables a direct comparison with the critical service application's

performance. The server component of the network monitor application then receives this channel condition time series and seamlessly adjoins it with the critical service performance time series. This integration facilitates a comprehensive, real-time visualization of the entire system, allowing for immediate correlation between network conditions and application performance. The collection and correlation of these metrics are fundamental to the benchmarking framework's ability to assess the resilience and efficiency of critical services operating over diverse wireless communication technologies and under varying network impairments.

Network Interfaces

This section details the implementation of both the wired and wireless network interfaces used in the benchmarking framework. The wired interfaces establish communication between the sensing node and its attached wireless modem, while the wireless interfaces connect the modem to the external network infrastructure, referred to as the black-box network. These components together form the communication backbone of the benchmarking system and represent one of the primary sources of reconfigurability for the experimental platform. By selectively enabling or disabling specific interfaces, the system can be adapted to operate across Wi-Fi 6, 4G LTE, and 5G mmWave environments with minimal software modification. The configuration of these interfaces defines the achievable throughput, latency, and reliability of the sensing node, directly influencing the measured performance metrics and overall experimental outcomes.

The network interfaces used throughout this work are shown in **Figure 3.15**, which depicts the physical devices and their connectivity to the sensing node. Each sensing node is equipped with a high-speed wired interface, a USB 3.2 Gen 1 Type-A connection, that links the processing platform to an external network adapter. This interface serves as a universal transport channel for data packets between the host system and the modem hardware, supporting up to 5 Gbps of full-duplex throughput. The use of USB as the primary interconnect ensures compatibility with a wide range of modular network adapters, facilitating rapid reconfiguration between experiments. It also enables the system to measure and compare the impact of distinct radio access technologies (RATs) under controlled and repeatable conditions.

Three primary wireless technologies were utilized to evaluate the performance of the benchmarking framework: Wi-Fi 6, 4G LTE, and 5G mmWave. Each technology represents a different generation of wireless evolution, offering distinct trade-offs in throughput, latency, and link stability. These wireless interfaces were selected not only for their relevance to modern edge

applications but also for their ability to span the spectrum of performance characteristics observed in real-world critical infrastructure deployments.

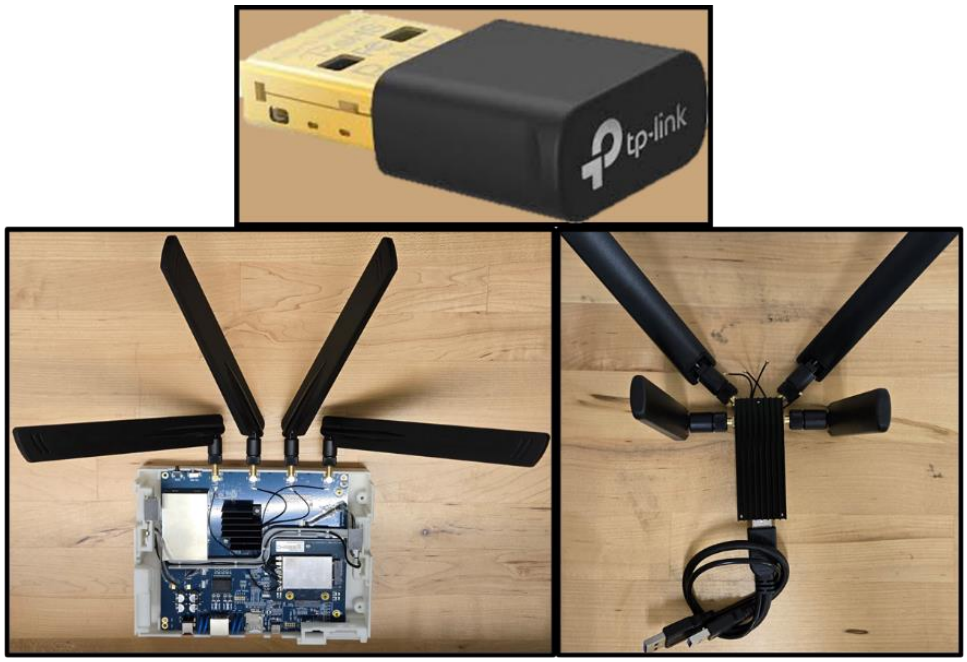

Figure 3.15: Network Interfaces used to Communicate with the System Controller. The devices depicted in this figure are network interfaces connected to the processing platforms of the sensing nodes via USB. The top image shows the Wi-Fi device TP-Link Archer T2U Nano. To the bottom left is the Quectel RM530N-GL mmWave Kit, and to the right is the Quectel RM530N-GL configured with a sub 6GHz USB module.

Wi-Fi

The first wireless interface implemented in this research is Wi-Fi 6, operating under the IEEE 802.11ax standard. The system employs a TP-Link Archer T2U Nano adapter, a compact 5 GHz USB wireless transceiver supporting the 802.11ac and 802.11ax protocols. In the benchmarking setup, the adapter was configured to operate exclusively in the 5 GHz band with a 1 Gbps channel bandwidth. This configuration allows for reduced interference, lower contention, and improved determinism compared to the crowded 2.4 GHz spectrum. Wi-Fi 6 introduces key physical- and MAC-layer enhancements—such as Orthogonal Frequency Division Multiple Access (OFDMA) and Multi-User Multiple Input Multiple Output (MU-MIMO)—which enable the system to maintain high throughput even under dense network conditions. These properties make it an ideal candidate for short-range, high-bandwidth edge scenarios such as those found in local-area industrial networks or on-premise surveillance systems. In this thesis, Wi-Fi 6 served

as an important middle ground between the ultra-reliable, high-capacity 5G mmWave link and the longer-range but lower-throughput 4G LTE configuration.

4G LTE

The second wireless configuration explored in this thesis is 4G Long-Term Evolution (LTE), implemented using the Quectel RM530N-GL cellular module, seen in **Figure 3.16**, configured for operation in the 1900 MHz (Band 2) frequency band. The LTE interface is connected to the sensing node through a USB 3.2 control and data interface, which allows for both AT command control and raw network data transfer. The module integrates a full 4G baseband stack compliant with 3GPP Release 16, providing an all-IP packet-switched connection between the user equipment (UE) and the Evolved Packet Core (EPC). The 4G configuration represents a widely deployed and mature communication technology, providing a stable yet relatively constrained link for real-time multimedia traffic. Its centralized scheduling model, based on Orthogonal Frequency Division Multiple Access (OFDMA) for the downlink and Single-Carrier Frequency Division Multiple Access (SC-FDMA) for the uplink, imposes latency limitations typically ranging from 30 to 100 ms under nominal conditions. In this benchmarking study, the 4G LTE configuration serves as a critical reference point for understanding the upper bound of latency and the lower bound of throughput achievable in wide-area wireless systems.

5G mmWave

The third and most advanced wireless interface implemented in this work is 5G New Radio (NR) operating in the millimeter-wave (mmWave) frequency range, specifically configured for 39 GHz (FR2 band n260) operation. The same Quectel RM530N-GL module was



Figure 3.16: Quectel RM530N-GL 5G Cellular Module. This device enables the sensing node cellular communication capability for both 4G LTE 1900MHz and 5G mmWave 39GHz bands. The module interfaces with a M.2 to USB carrier board to communication with the host system. It also connects with sub-6 and mmWave antenna via IPX coax cables.

used with a dedicated mmWave development kit. This configuration enables access to extremely wide bandwidths, offering theoretical peak data rates well above 2 Gbps and air-interface latencies below 10 ms under optimal conditions. The module interfaces with the sensing node via USB 3.2 for control signaling and data exchange, while the radio front-end connects to the 5G base station (gNodeB) through its RF subsystem. The 5G mmWave configuration was deployed in the AT&T 5G MEC Lab at the University of Tennessee, Knoxville, which provided a controlled testbed environment for high-frequency cellular experimentation. Among the three wireless technologies evaluated, 5G mmWave exhibited the most significant sensitivity to physical factors such as line-of-sight blockage and signal attenuation, but also demonstrated the lowest achievable latency and highest throughput in optimal conditions.

Loopback

To establish a baseline and isolate the computational performance of the sensing nodes from network-induced effects, a loopback configuration was implemented. In this mode, network packets are transmitted and received internally within the same device through its software-defined network stack, bypassing the physical network interface entirely. This configuration effectively removes all external variables associated with the radio channel, enabling pure measurement of application-layer processing and system-level latency. Loopback results serve as an idealized control condition for benchmarking experiments, allowing for direct comparison of device performance independent of wireless impairments or bandwidth limitations.

Collectively, these network interfaces form a modular and reconfigurable communication architecture that allows the benchmarking framework to evaluate performance across diverse network conditions. By combining wired USB interfaces with interchangeable wireless adapters, the system enables precise control over physical- and data-link layer characteristics while maintaining compatibility with common edge computing hardware. The comparative analysis of Wi-Fi 6, 4G LTE, and 5G mmWave links—alongside the loopback baseline—provides a holistic view of how network design choices and physical constraints influence the performance of latency-sensitive applications deployed at the network edge.

Cyber-physical Space and Realistic Operating Conditions

A crucial aspect of the system design and implementation is the cyber-physical space lying between the sensing node's wireless modem and the radio interface of the black-box

network. In this space, several permutations of operating conditions are enforced to alter the condition of the communication channel. One example of this is the use of reflectors within the wireless environment to convolute the path between radio devices. This enables the benchmarking framework to analyze the system performance while under realistic conditions intended to degrade the performance of the streaming application. Ultimately, we test the network's ability to counteract non-ideal effects due to degradation and explore the applications response to the observed network changes.

Conclusion

This chapter translated the methodological framework developed earlier into a fully realized benchmarking system that integrates sensing, networking, and control components within a modular, reconfigurable architecture. The resulting platform establishes an end-to-end measurement chain capable of observing real-time application performance in conjunction with wireless link behavior. By structurally separating the sensing node, network layer, and system controller, the design enables precise attribution of observed effects to either computational or transport-layer phenomena while maintaining flexibility to accommodate diverse experimental configurations. This separation of concerns ensures that network dynamics, codec behavior, and inference workloads can be independently varied and rigorously compared under consistent conditions.

At the implementation level, the use of Linux V4L2 and GStreamer provides a media pipeline with access to buffers, encoder states, and timing hooks essential for performance instrumentation. The inclusion of an RTP header extension for per-frame timestamping enables direct measurement of one-way delay, while synchronized telemetry logging aligns application-level quality metrics with radio and channel indicators in real time. Together, these features form a comprehensive observability plane that unites live visualization, automated data collection, and post hoc analysis. The modular USB-based transport interface further extends the system's versatility, allowing identical workloads to be evaluated across Wi-Fi 6, 4G LTE, and 5G mmWave radios, as well as a local loopback control path that isolates compute-only performance.

The chapter also demonstrated how precise temporal coordination preserves the validity of per-frame latency and jitter estimates. This attention to timing discipline, combined with a dedicated impairment environment, allows the benchmark to move seamlessly between ideal and

degraded conditions without compromising measurement integrity. Collectively, these design choices yield a reproducible, cross-layer testbed that captures both the system's computational workload and its wireless context, addressing the core benchmarking requirements identified in Chapter 2.

With the system architecture now established, Chapter 4 formalizes the experimental methodology that governs its use. The next chapter defines the constants, parameters, and procedural steps necessary to execute controlled trials across the various radio access technologies. It details the baseline and impaired configurations, outlines the statistical treatment of synchronized metrics, and specifies how workload, codec, and network factors are isolated. Together, these elements ensure that the results presented in the subsequent chapters reflect intrinsic properties of the system's performance, rather than artifacts of the measurement process itself.

CHAPTER FOUR
EXPERIMENTAL METHODOLOGY

This chapter details the methods and procedures employed to conduct the experiments and collect results. The description begins with the physical setup of the processing and network devices, followed by a discussion of the performance metrics captured during each trial. The network conditions and operational environment are then outlined to establish the context under which experiments were performed. Finally, the procedures for executing experiments, collecting data, and performing post-analysis are presented. Together, these components define the experimental methodology, providing a clear account of the physical layouts, conditions, and environments used to obtain the results discussed in subsequent chapters.

Experimental Setup

This section details the physical connections and configurations assembled for the benchmarking system's experimental runs. As outlined in Chapter Three, the system comprises three primary components: a sensing node, a system controller, and the black-box network. The sensing node acts as an edge device, capturing video and performing a processing task. Concurrently, the system controller receives the streamed video, measures key performance metrics, and provides a real-time display of both video and metrics. The experimental setups are organized by their network mode—baseline, Wi-Fi, 4G, and 5G—and each configuration specifies the processing platform for the sensing node, the modem for network access, any additional networking equipment, and the parameters held constant between experiments.

Constants

The experimental setup maintains several parameters constant across all trials to ensure consistency and comparability of results. These fixed components include the streaming and analysis programs (i.e., the base workload), the video source, and the operating environment. Standardizing these parameters enables a more reliable correlation between benchmarking outcomes. Specifically, the encoding algorithms perform under identical conditions, as the same video source is looped continuously every minute. Likewise, all processing platforms execute the same base workload, minimizing variations in performance that could arise from differences in system or kernel-level capabilities. Additionally, the radio frequency (RF) environment remains consistent across experiments by maintaining an identical configuration and quantity of RF equipment within the testing laboratory.

Baseline

The baseline configuration is vital to this study as it allows for the isolation of networking factors such as propagation delay, buffering, and congestion. Furthermore, it allows for a non-network biased result, where the processing power, application algorithms, and protocols can be analyzed and later compared to the wireless setups. The baseline configuration, shown in, details how the processing platform connects to the video source and the path that video data takes.

The baseline configuration is crucial to this study because it isolates networking factors like propagation delay, buffering, and congestion. This setup provides results that are not biased by network performance, allowing for a focused analysis of the system's processing power, application algorithms, and protocols. As shown in **Figure 4.1**, the baseline configuration details the physical connection between the processing platform and the video source, illustrating the complete data path.

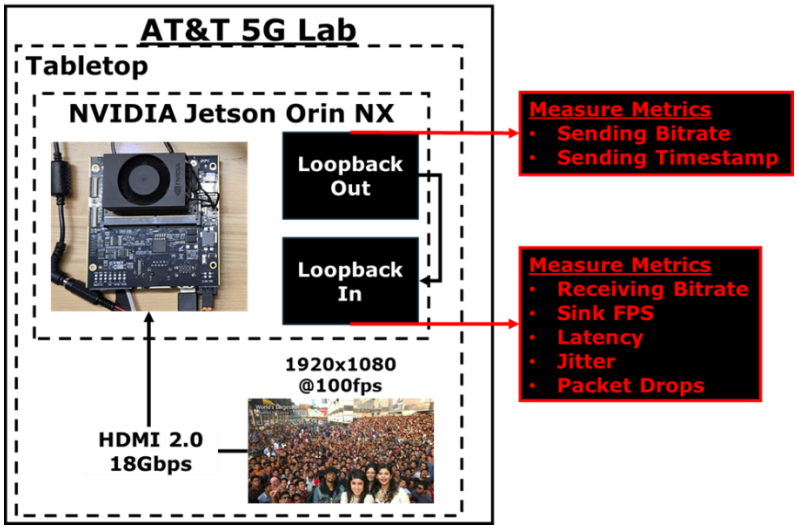

Figure 4.1: Baseline Network Configuration. This diagram details the network configuration used as a baseline for comparing wireless configurations and processing platforms. It utilizes the device's internal network interface known as loopback. The benefit of this configuration is that it can negate the constricting effects of a wireless network. Furthermore, it enables us to isolate the devices processing power for an apples-to-apples comparison between devices.

The loopback interface avoids the negative effects of networking factors by relying on the Linux kernel to forward packets back to itself. Because all data remains in memory, it can be accessed almost instantly by the terminal socket. This process completely bypasses the network handover, over-the-air propagation, and routing that are inherent to wireless networks like Wi-Fi.

For this thesis, this setup allows us to measure the system's performance based purely on the device's capabilities. The influential factors in this configuration are memory speed, memory size, and processing capacity. This approach establishes a baseline performance for the devices and the application, which will later be used to analyze performance changes in wireless configurations.

Wi-Fi

The first of three wireless setups outlined in this chapter is Wi-Fi 6. While the general networking structure of each configuration is very similar, the specific devices and radio technology vary greatly. As seen in **Figure 4.2**, the processing platform is outfitted with a TP-Link Archer T2U Nano Wi-Fi 6 modem. This device allows for wireless communication to the MEC compute resources via a Ericsson Wi-Fi 6 gateway and router and a wired fiber optic link. The modem and router use only the 5GHz carrier.

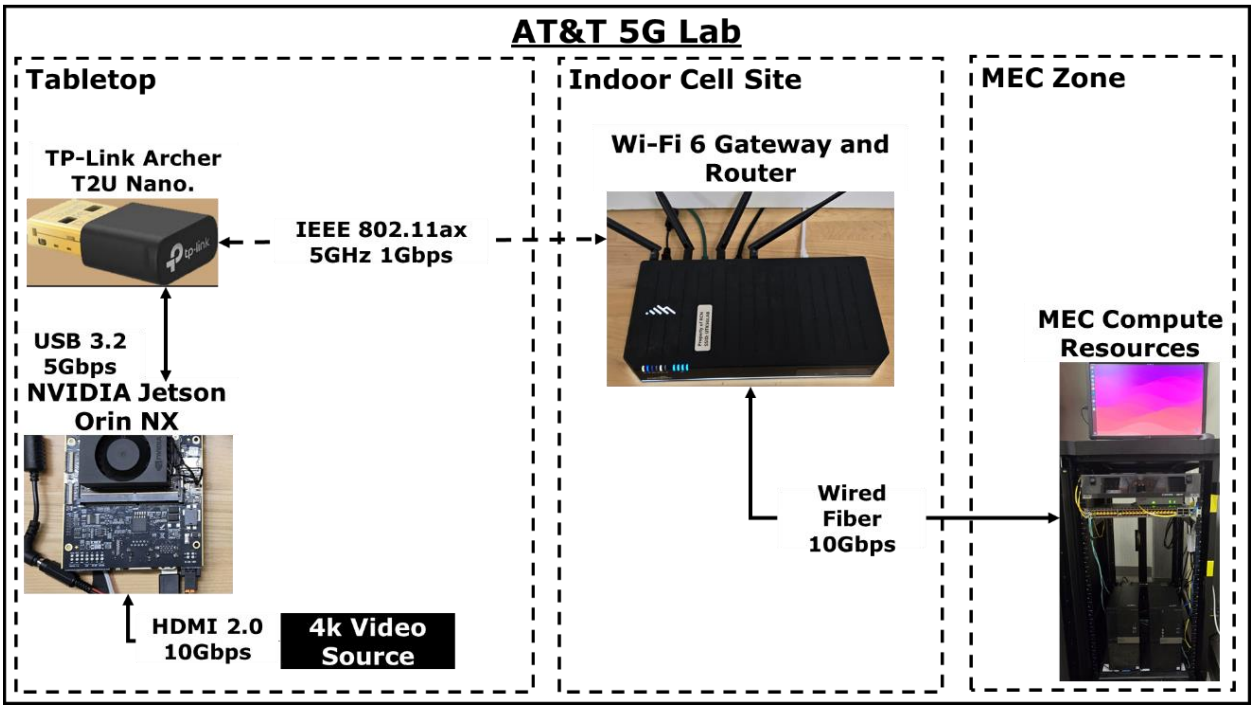

Figure 4.2: Wi-Fi 6 Network Configuration. The physical setup of the Wi-Fi network configuration is detailed in this diagram, which is organized into three distinct sections. The sensing node system is shown on the far left, positioned on a tabletop inside the AT&T 5G Lab at UTK. The middle section depicts the Wi-Fi router and modem on a separate tabletop, acting as an intermediary between the sensing node and the third section, the MEC Zone. The sensing node transmits its video stream to the Wi-Fi router over an IEEE 802.11ax 5GHz wireless channel with a 1 Gbps bandwidth. Upon receiving the data, the router forwards the video packets to the system controller, located in the MEC Zone, via a 10 Gbps fiber optic cable.



Figure 4.3: Wi-Fi 6 Ericsson CradlePoint Router. This wireless gateway router operates as the access point for the sensing nodes and system controller. It enables Wi-Fi 6 5GHz communication between experimental devices.

4G

The second wireless network configuration to discuss is the cellular setup with 4G LTE. Again, data captured from the sensing node is transmitted over a wireless link to the intermediate network infrastructure and then forwarded along a wired link to the system controller in the MEC Zone. This configuration can studied from **Figure 4.4**, where the processing platform can be seen connected to a cellular modem. Due to the non-standalone network type constrained by the internet service provider, both the 1900MHz and 39GHz carriers are present, however, once video begins streaming, the modem strictly utilizes the high-frequency channel.

Within the UTK/AT&T 5G lab, the cellular base station is outfitted with two radio and antenna components, one for 4G 1900MHz and the other for 5G+ mmWave at 39GHz. These access points are directly wired to the digital converter modules of the cell site.

To disable the use of 39GHz for the 4G experiments, the cellular modem configuration is altered with a kit lacking the necessary antenna as shown in **Figure 4.6**. The underlying modem device remains the same, as well as the wired interface to the processing platform.

5G

The final wireless configuration used in the experimentation of this thesis is cellular 5G New Radio (NR). This setup is nearly an exact copy of the 4G setup however the wireless channel and radio device, shown in **Figure 4.5**, are different.

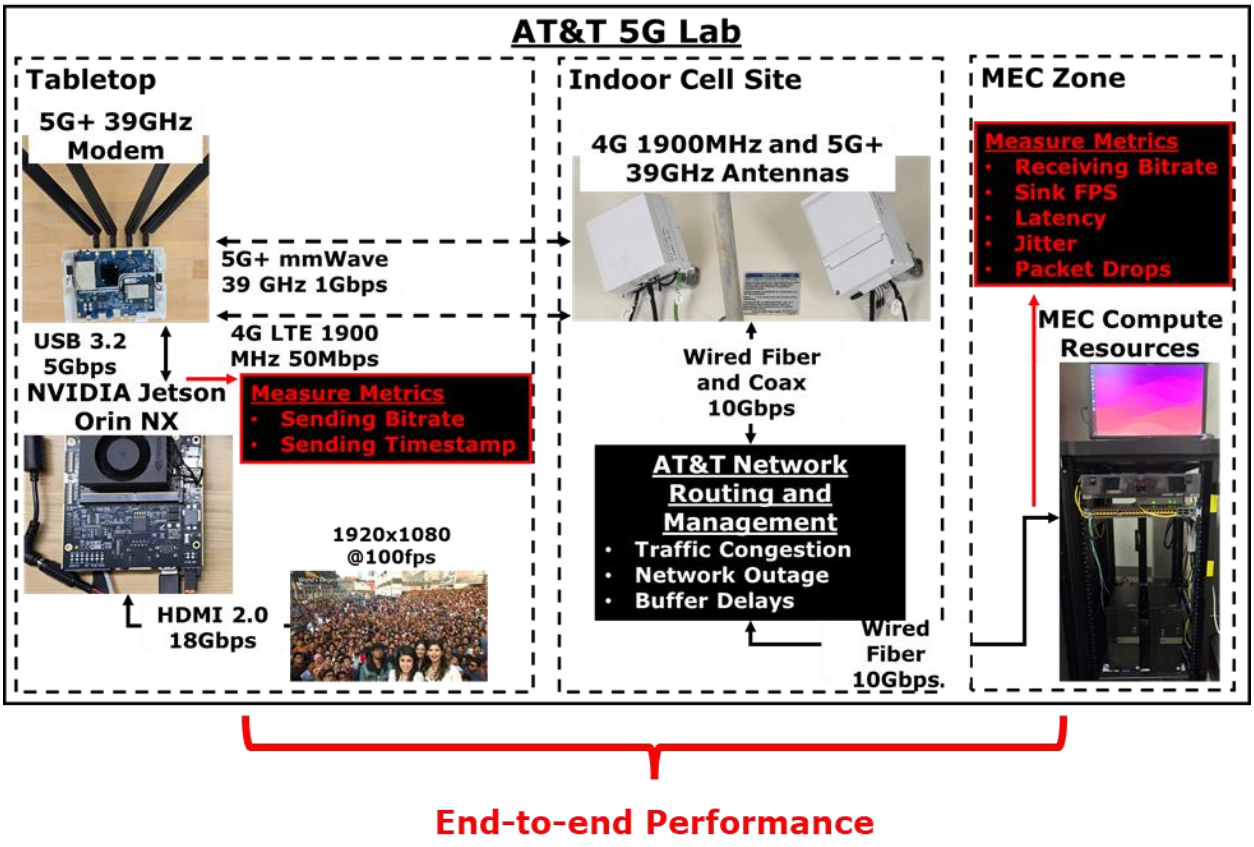

Figure 4.4: Cellular Network Configuration. Both the 4G and 5G wireless modes utilize this setup, however, for the former, the mmWave antenna are disengaged to prohibit use of 39GHz.



Figure 4.5: AT&T 5G Lab Antenna and Radio Units. These units enable the cellular service to the user equipment (UE) operating in the lab. For this thesis, these UE are literally the sensing nodes which capture and stream video to the MEC resources within the cell site. The 4G LTE radio on the left operates at a frequency of 1900MHz and utilizes a massive input massive output (MIMO). The 5G mmWave radio on the left operates at a frequency of 39GHz and utilizes a massive input massive output (MIMO). Both units forward their upstream data to the cell sites baseband units via a coaxial cable.

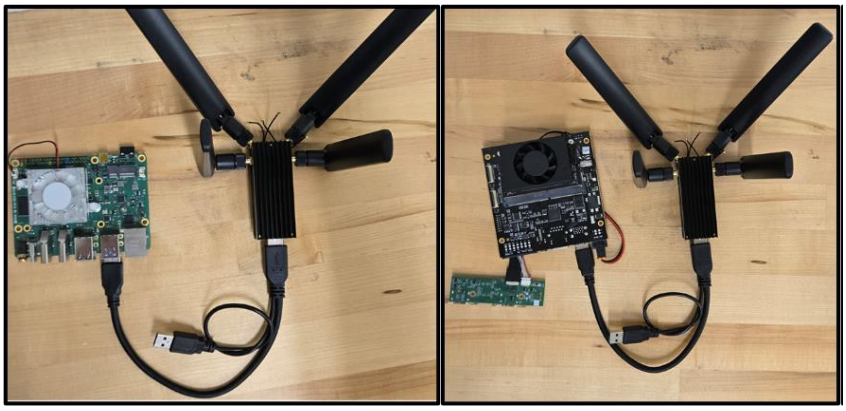

Figure 4.6: Processing Platform and 4G Cellular Configuration. This figure depicts three processing platforms, namely the Radxa Rock 5B on the left and Jetson Orin NX on the right, connected via USB 3.2 to a Quectel RM530N-GL 4G Cellular module. These devices compose the edge sensing node's processing platform. Similar configurations are assembled for wireless modes Wi-Fi 6 and 5G+ mmwave.

Performance Metrics

The benchmarking system designed and implemented in this thesis, probe and measure aspects of the underlying network infrastructure and the capabilities of the hardware and software components of the sensing node and system controller. The end-to-end metrics include, received bitrate, one-way-delay (latency), frame rate of the displayed video, network packet loss rate, and network jitter. These metrics allow the benchmarking framework to analyze how well the underlying network supports a multimedia load.

Conclusion

This chapter established the experimental methodology that transforms the conceptual system architecture into a repeatable and functional benchmarking framework. Through strict control of workloads, video sources, and RF conditions, the protocol ensures that performance variations can be directly attributed to network or computational factors rather than environmental variability. The inclusion of four standardized modes—loopback, Wi-Fi 6, 4G LTE, and 5G mmWave—enables consistent cross-mode comparison and clear attribution of performance effects. Each configuration was selected to represent a distinct operational regime, from deterministic LAN behavior to the variability of wide-area cellular channels, while maintaining hardware and interface consistency across all trials.

By integrating synchronized logging of both application-level KPIs and radio telemetry, the framework provides a causal view of performance, connecting fluctuations in bitrate, latency, or jitter with underlying channel and compute dynamics. The deliberate inclusion of controlled

impairments extends the benchmark's scope to evaluate system stability and resilience under non-ideal conditions, providing insight into transient and tail behaviors relevant to mission-critical applications. Together, these design choices yield a controlled yet realistic test environment that balances reproducibility with operational fidelity.

The result is a defensible, end-to-end benchmarking methodology that distinguishes intrinsic system limits from network-induced effects and quantifies how codec, processing, and RAT selection shape real-time performance. This foundation supports the subsequent analytical work in Chapter Five, where empirical results are used to characterize the performance envelopes of each configuration and validate the framework's capacity for causal and comparative analysis.

CHAPTER FIVE
RESULTS AND DISCUSSION

The results presented in this section summarize the experimental evaluation conducted across a structured matrix of test conditions, varying along four primary dimensions: device, wireless mode, detection, and encoding. Specifically, performance was measured on two hardware platforms—NVIDIA Jetson Orin NX and Radxa Rock 5B—under three wireless modes: loopback, Wi-Fi 6, and 4G LTE. Each trial was further tested with and without detection enabled, and with video encoding using either H.264 or H.265. Plots of the resulting trials are presented to highlight differences in device capabilities, the relative impact of wireless network conditions, and the trade-offs introduced by detection and encoding configurations. The following discussion interprets these results with respect to both hardware performance and network behavior, with an emphasis on identifying patterns and bottlenecks relevant to real-time streaming applications.

Performance Across Processing Platforms

This section compares the performance of two distinct processing platforms, the Jetson Orin NX and the Radxa Rock 5B, to evaluate the role of hardware architecture in supporting real-time video streaming at the edge. The analysis focuses on how each platform's bitrate, latency, and framerate are affected by the application of a computationally intensive task. The results will first be presented on a per-device basis, followed by a summary that quantifies the relative performance gain of one platform over the other.

Raw Data Logged by System Controller

The following plots in **Figure 5.1** and **Figure 5.2** show a performance comparison of two processing platforms using several key metrics. The trials were conducted with the following constants: H.265 encoding, a 1080p video source, and a shared cellular network environment. Each platform underwent four 30-minute trials during which it captured video, optionally applied the RetinaFace model, and streamed the encoded video. The sampling rates varied between metrics, with bitrate having the highest precision (approx. two million samples) and latency having a much lower sample count (approx. 24,000 samples). Furthermore, to facilitate a direct comparison, each trial's time series data was aligned so that the initial timestamp corresponded to time zero on the x-axis. The plots allow for a relative performance assessment between the two

platforms by comparing key performance metrics (KPMs) when detection is enabled versus disabled.

The first results presented in this section were collected using a setup where a Radxa Rock 5B served as the sensing node. This node transmitted video either to the system controller via a 4G network or back to itself using a loopback interface, with H.265 used for encoding and compression. The key metrics—bitrate, latency, and framerate—were captured during this trial and are presented in **Figure 5.1**. The findings indicate that enabling additional processing has a significant impact on all three metrics.

The act of enabling detection on the Radxa Rock 5B has a profound impact on system performance. As a direct result of the processing time needed to extract facial features and overlay bounding boxes from the ML model, the bitrate is reduced by a factor of three, latency decreases by approximately 30%, and the framerate drops by over four times. This increased processing time results in a larger inter-frame delay, which slows the rate of video transmission and reduces the total amount of data that occupies the link at any given time. Consequently, this method of complex processing may be an unacceptable trade-off when a consistent, high bitrate is a primary concern.

After switching the configuration to substitute the processing platform from the Radxa Rock 5B to the NVIDIA Jetson Orin NX, a similar trend is seen. Again, the bitrate, latency, and framerate were captured while streaming H265 video to the system controller over 4G or to itself using the loopback interface. These results are shown in **Figure 5.2**.

Using the NVIDIA device as the sensing node, we observed that enabling detection caused a twofold reduction in bitrate, a 3.5x reduction in latency over a 4G link, and a decrease in framerate by more than half. While these findings still support the conclusion that additional processing reduces the volume of transmitted video, the results show that the more powerful NVIDIA device introduces a considerably smaller performance impact compared to the Radxa device.

Post Analysis Probability Distribution

To dive deeper into the effect of processing power and load on video streaming performance, a collection of plots are presented showing the probaility density function (PDF) and cumalitive distribution function (CDF) of the KPM. Additionally, the plots provide a legend

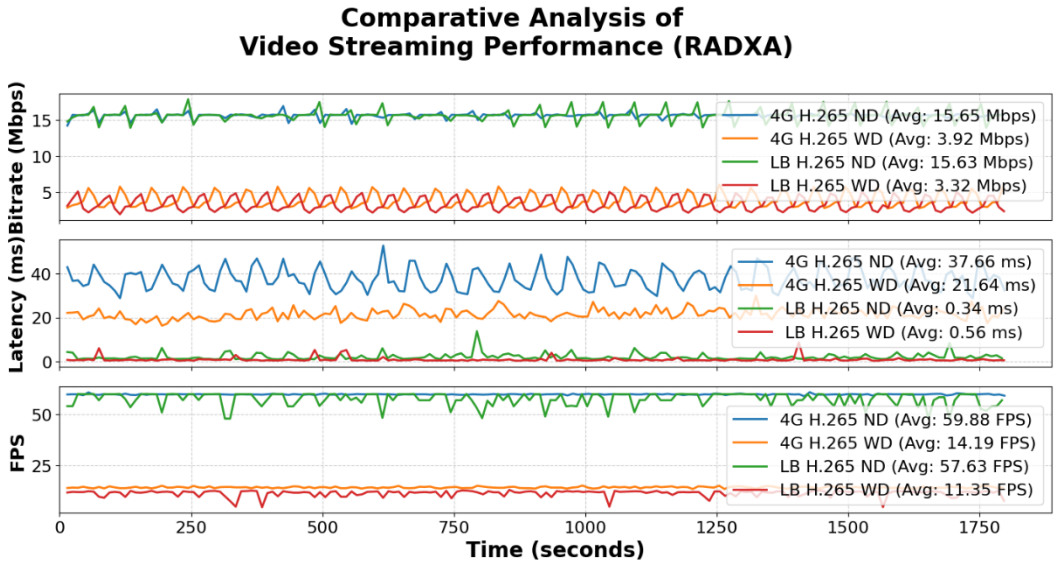

Figure 5.1: Comparative Analysis of Video Streaming Performance (RADXA). This figure presents the performance of the Radxa Rock 5B across three key metrics: bitrate, latency, and framerate. The trials were conducted under two variable conditions. The first is the network configuration, comparing a cellular 4G AT&T network (1900 MHz band) with a loopback configuration (LB), which uses the device's internal network interface. The second variable is the application of on-device processing, specifically facial feature extraction using the RetinaFace model. Each plot's legend includes the global average of the raw time series.

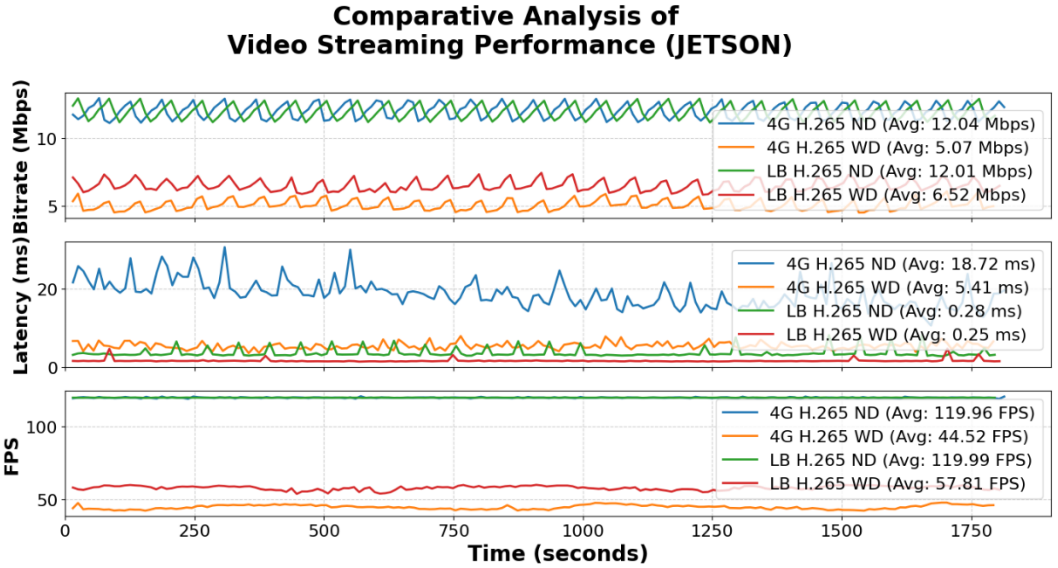

Figure 5.2: Comparative Analysis of Video Streaming Performance (JETSON). This figure presents the performance of the NVIDIA Jetson Orin NX across three key metrics: bitrate, latency, and framerate. The trials were conducted under two variable conditions. The first is the network configuration, comparing a cellular 4G AT&T network (1900 MHz band) with a loopback configuration (LB), which uses the device's internal network interface. The second variable is the application of on-device processing, specifically facial feature extraction using the RetinaFace model. Each plot's legend includes the global average of the raw time series.

containing the median, 95th percentile and standard deviation. The plots are dived by processing platform and metric. **Figure 5.3**, dispalys the bitrate metric captured from the Radxa configuration, and the following **Figure 5.4** presents the results from the NVIDIA configuration. These plots give key insight into the stablity and reliability of these configurations.

While the Radxa device provided a slightly higher overall bitrate, the NVIDIA device demonstrated greater reliability. This is evidenced by the average 17 Mbps bitrate with a standard deviation of 1.7 for the Radxa (without detection enabled), compared to the NVIDIA's 13 Mbps bitrate with a standard deviation of 0.7. When detection is enabled, both devices experience a reduction in overall bitrate and a slight decrease in performance reliability.

Figure 5.5 and **Figure 5.6** present the PDF and CDF of the framerate metric for the Radxa and NVIDIA devices respectively. To reiterate, these results are taken from four runs of the benchmarking application on each device. Two runs are conducted over a 4G link with and without detection enabled, and two runs coducted over the loopback interface.

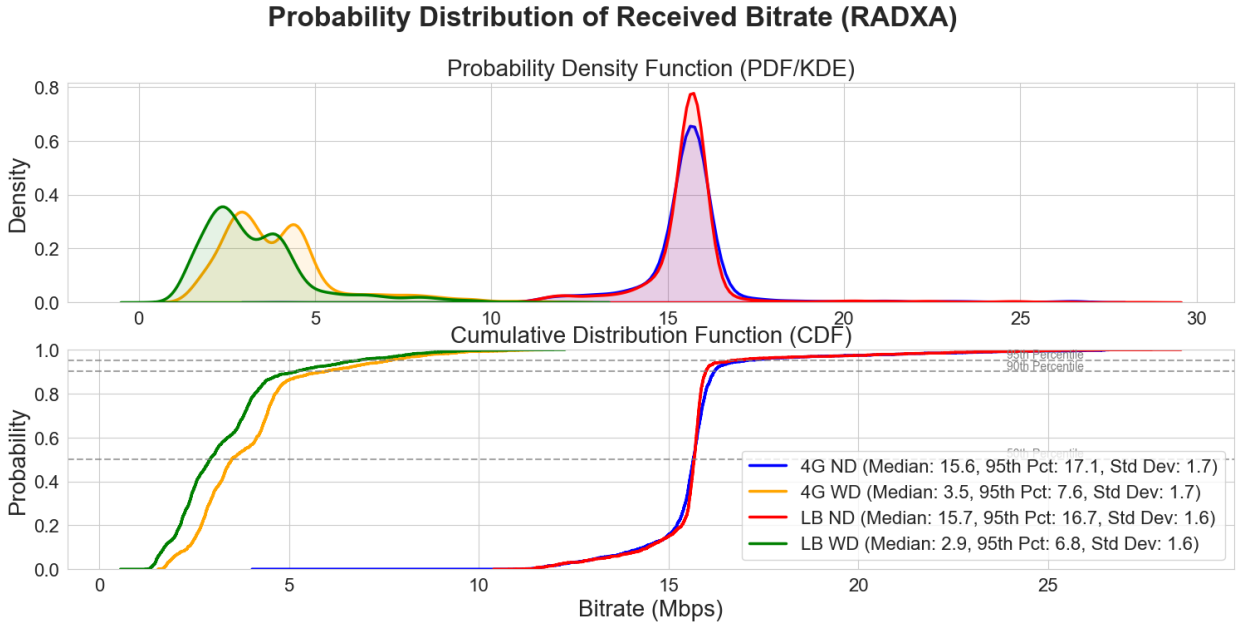

Figure 5.3: Probability Distribution of Received Bitrate (RADXA). The plots displayed here summarize the distribution of the video stream's received bitrate for the Radxa Rock 5B. They highlight two primary findings. The scenarios without object detection consistently achieve a bitrate ~4.5 times greater than that with detection enabled, and enabling detection introduces significant performance variability, as shown by the much wider spread of the PDFs. The minimal difference between the 4G and loopback traces suggests that, for this application, network conditions were not the primary performance bottleneck.

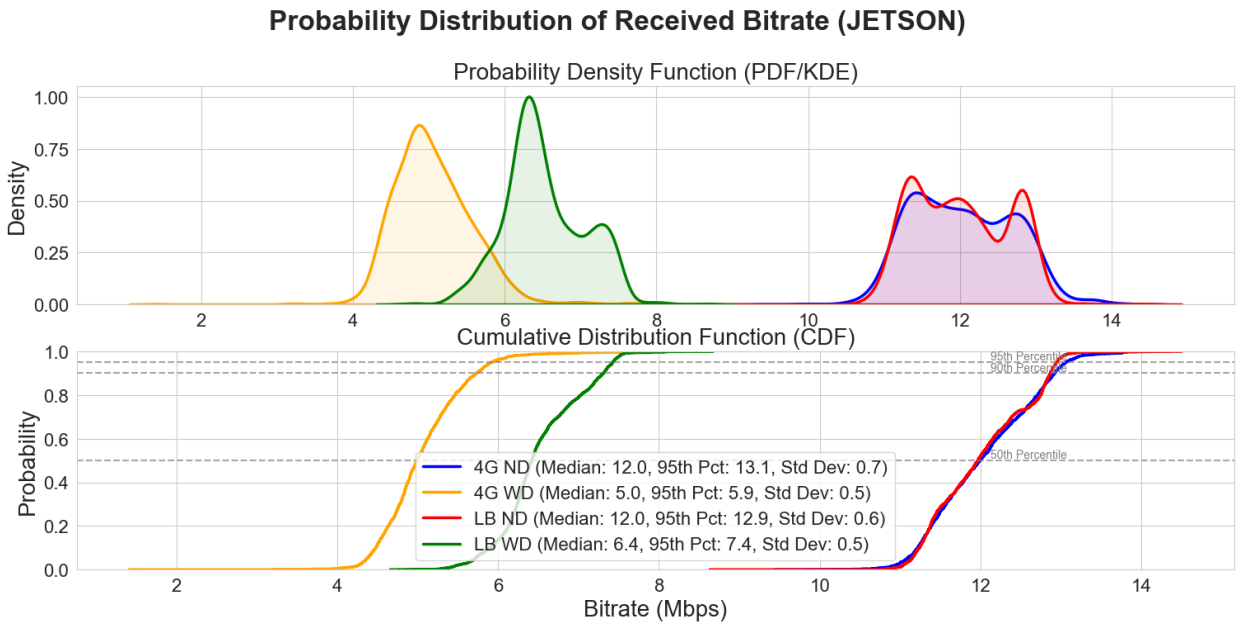

Figure 5.4: Probability Distribution of Received Bitrate (JETSON). The plots displayed here summarize the distribution of the video stream's received bitrate for the NVIDIA Jetson Orin NX. Like the Radxa device, traces captured from the tests without detection enabled achieved a increase in bitrate two times greater than that with detection enabled. However, there was approximately a 1 Mbps increase when transmitting across the loopback interface with detection enabled. Furthermore, the 95th percentile was 1.25 times greater than that using 4G.

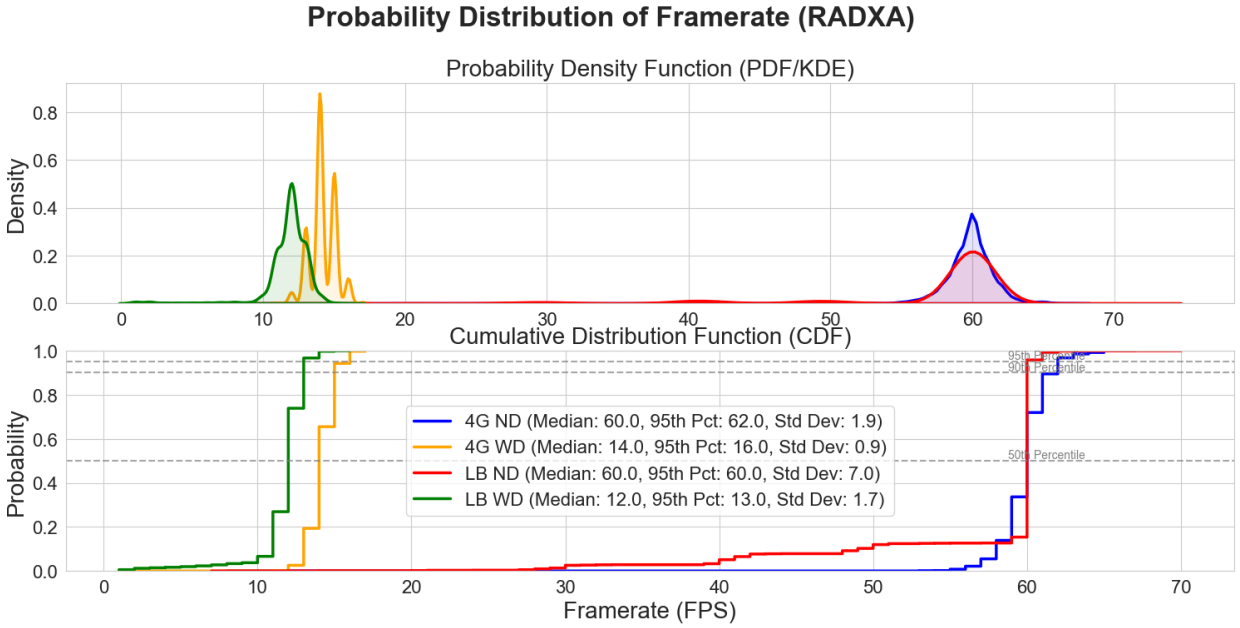

Figure 5.5: Probability Distribution of Framerate (RADXA). These plots display the PDF and CDF of framerate seen by the system controller when transmitted from the Radxa Rock 5B. A similar trend continues of greater bitrates for tests without detection enabled. For these traces, there was a ~4.3 times increase between detection enabled vs. disabled.

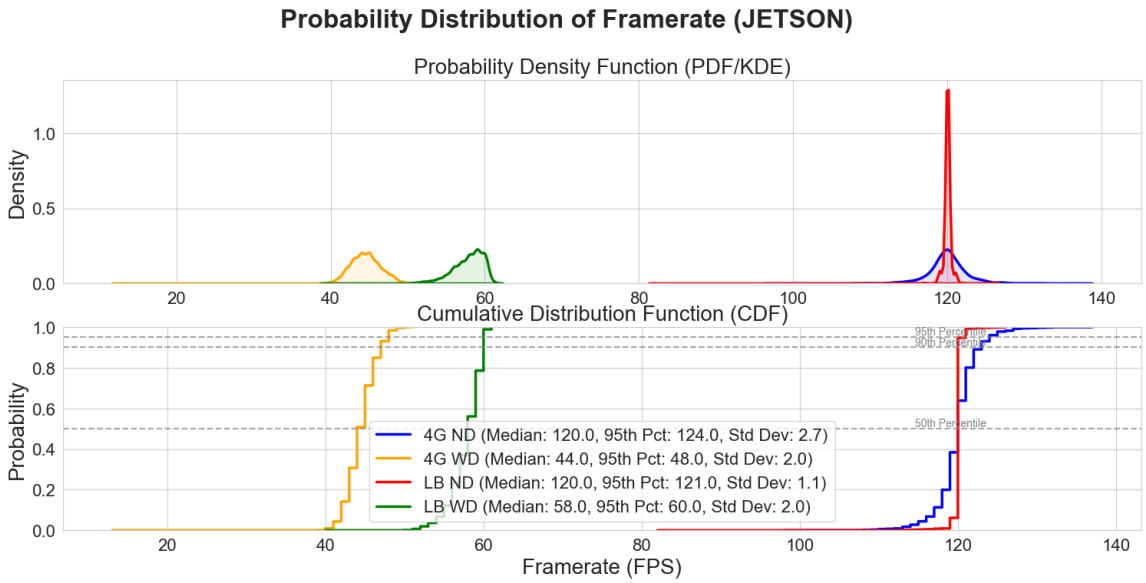

Figure 5.6: Probability Distribution of Framerate (JETSON). These plots display the PDF and CDF of framerate seen by the system controller when transmitted from the NVIDIA Jetson Orin NX. A similar trend continues of greater bitrates for tests without detection enabled. For these traces, there was a ~2-3 times increase between detection enabled and disabled.

The Radxa device demonstrated its highest performance with the loopback interface and detection disabled. The 95th percentile reached 60 fps but experienced a higher standard deviation than with detection enabled. This is especially apparent in the loopback interface where it suffers a decrease in reliability by nearly a factor of seven. In comparison, the NVIDIA device also demonstrates its best performance of 124 fps while transmitting over 4G with detection disabled. There is an obvious reduced impact on reliability when disabling detection on the NVIDIA device. The standard deviation reached 2.7 without detection over 4G but only reached two when detection was enabled.

The PDF and CDF are also presented for the latency metric as seen in **Figure 5.7** and **Figure 5.8** of the Radxa and NVIDIA configurations respectively. These results give key insight into the effect of additional processing on the end-to-end delay.

In the loopback configuration, the near-zero propagation delay results in little to no difference in latency between enabling or disabling detection. In contrast, the 4G link shows a significant increase in latency when detection is disabled for both devices. The Radxa device experiences a 2.4x increase in delay, while the NVIDIA device sees a 3x increase. Disabling detection also negatively impacts reliability, as indicated by a 2.7x increase in standard deviation for the Radxa device and a 2.3x increase for the NVIDIA device.

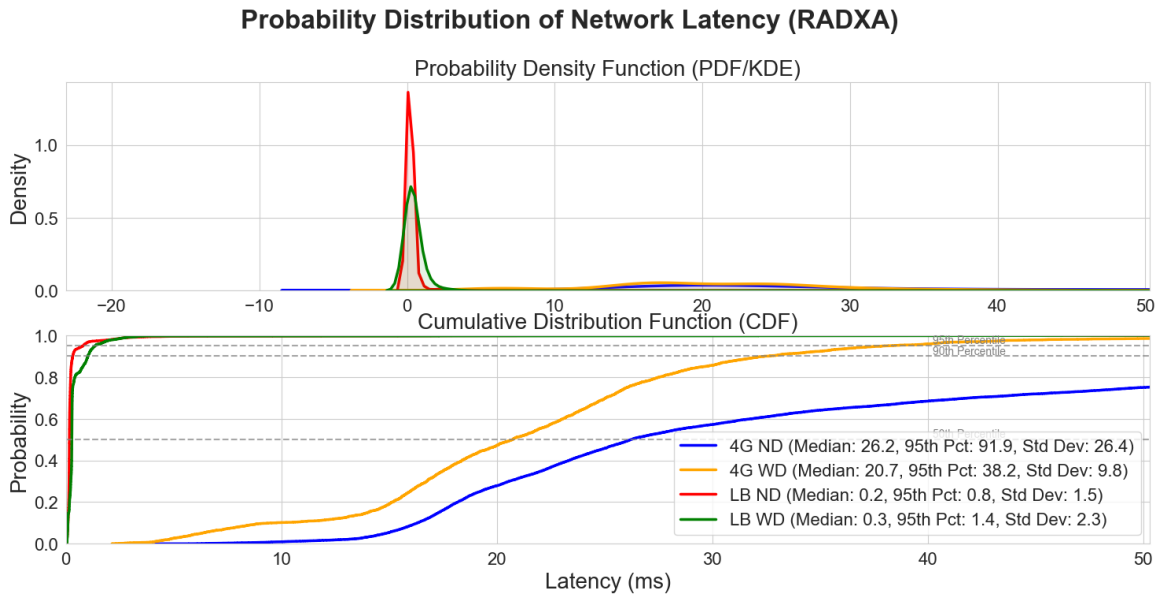

Figure 5.7: Probability Distribution of Network Latency (RADXA). The PDF and CDF of the end-to-end latency between the Radxa Rock 5B and system controller give further insight into the performance of this configuration. Two key findings can be extracted. The first surrounds the network link, where 4G performance suffers greatly. Not only is there nearly a 100x increase in latency, but there is also significant variation. The second key finding can be seen by comparing the cases with detection enabled to those without. Enabling detection results in a decrease of variation, namely over a two times greater deviation for the 4G case and 1.5x greater for the loopback case.

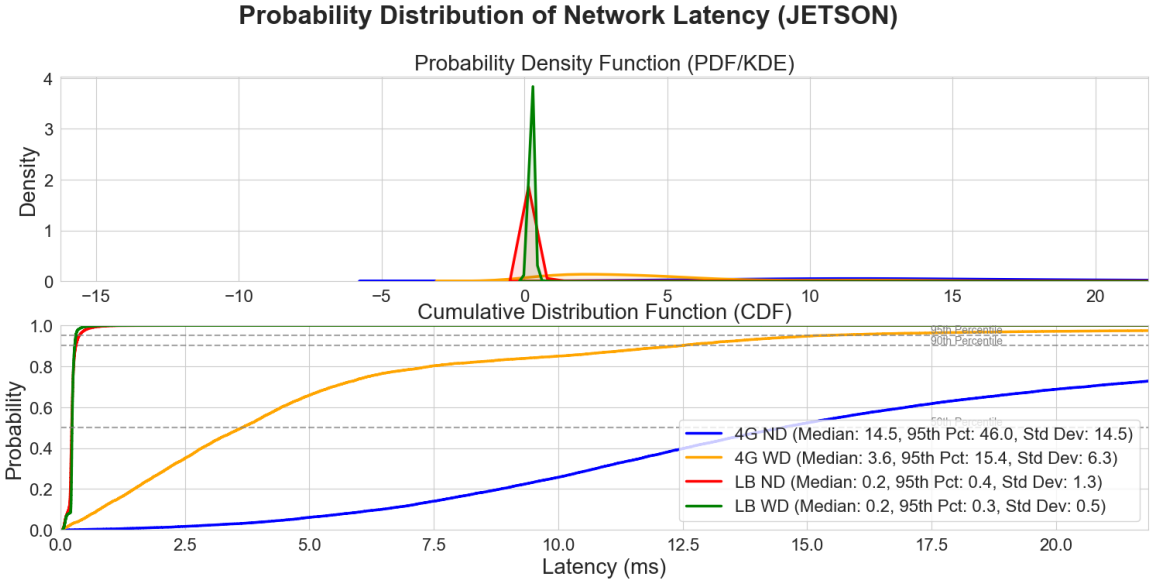

Figure 5.8: Probability Distribution of Network Latency (JETSON). The PDF and CDF of the end-to-end latency between the NVIDIA Jetson Orin NX and system controller give similar insight into the streaming performance as the Radxa device. Again, the latency is much more consistent and lower than the 4G cases, and the deviation is significantly increased when disabling detection.

Performance comparisons across different devices with varying processing loads provide quantifiable evidence of a strong, consistent relationship between the load and performance across all three KPMs.

As a general trend, the received bitrate of the video stream is significantly reduced when detection is enabled due to the additional delay injected between video frames. The framerate is directly subject to the bitrate and is therefore reduced by a nearly equal factor. With less competing traffic on the network, data is buffered for a shorter duration, leading to shorter end-to-end delays.

The results provided demonstrate the benchmarking application's effectiveness in capturing the influence of hardware architecture on real-time video streaming performance under computational load. By comparing the Radxa Rock 5B and NVIDIA Jetson Orin NX, the results reveal a strong correlation between processing capability and key performance metrics—bitrate, latency, and framerate—when an intensive ML-based detection task is introduced. Across all experiments, enabling the RetinaFace model consistently reduced bitrate and framerate while increasing latency, confirming that added processing delays directly hinder streaming efficiency. However, the magnitude of these effects varied sharply between platforms. The Radxa device exhibited greater performance degradation and variability, indicating limited processing headroom and higher sensitivity to concurrent workloads. In contrast, the Jetson Orin NX maintained higher stability and smaller deviations across metrics, highlighting its architectural advantages such as hardware acceleration and larger GPU capacity. The probability and cumulative distribution plots further reinforced these findings by quantifying the stability and predictability of each configuration—showing tighter distributions and lower standard deviations on the NVIDIA device. Collectively, these results emphasize the critical role of hardware resources in sustaining low-latency, high-throughput video delivery at the network edge, aligning with the thesis objective of establishing standardized, comparative methods to evaluate real-time networked systems under varying computational and communication constraints.

Performance Across Wireless Technologies

This section compares the performance of three wireless technologies—cellular links using 4G and 5G, and Wi-Fi 6—to evaluate the role of network capabilities in supporting real-time video streaming at the edge. The analysis focuses on how effectively the network facilitates communication between the client and server of the benchmarking application. Results are first

presented on a per-device basis for tests configured with the H.264 and H.265 encoders, followed by an analysis of the probability distributions of key performance metrics.

Raw Data Logged by System Controller

The plots in figures, **Figure 5.9**, **Figure 5.10**, **Figure 5.11**, and **Figure 5.12** display the raw time series data logged by the system controller during experiments conducted over three wireless modes. To reduce noise and improve visibility, the values are averaged over discrete time bins. Each figure includes plots for bitrate (Mbps), latency (ms), frame rate (FPS), packet loss rate (packets lost over total packets sent), and network jitter (variation in delivery times, ms). Legends within the plots indicate averages computed over the entire experiment. Each experiment was conducted for approximately thirty minutes.

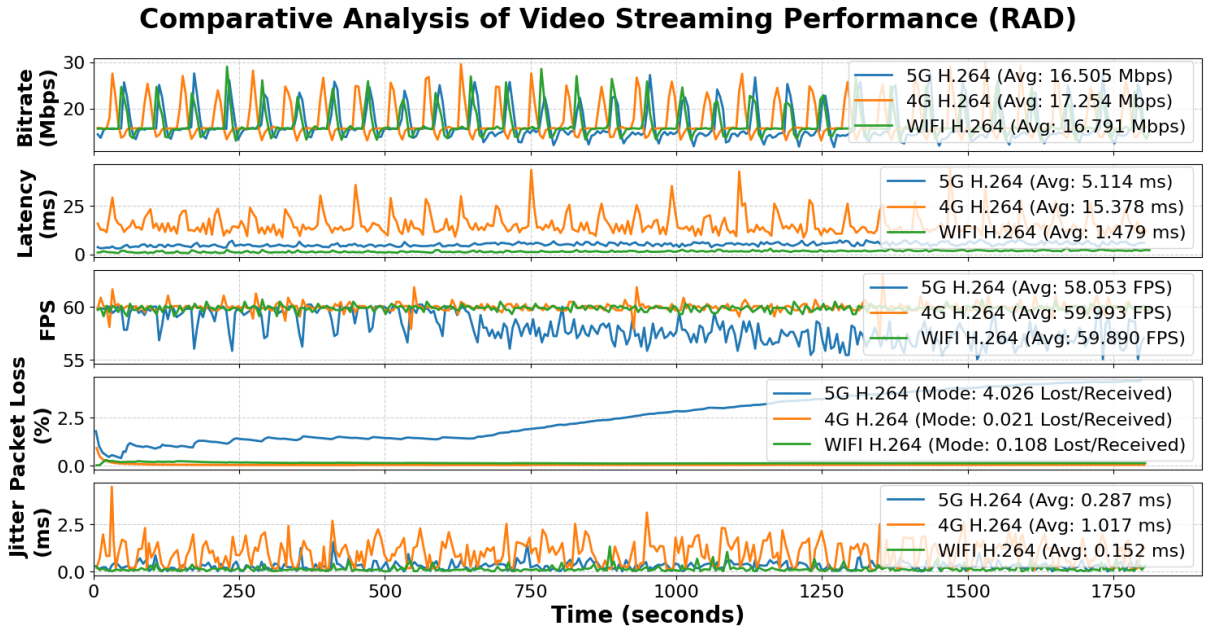

Figure 5.9: Raw Streaming Performance Data Across Wireless Technologies (Radxa/H.264). This figure displays a post analysis of the raw metrics logged by the benchmarking application run on the Radxa Rock 5B with the H.264 encoder enabled. It includes plots of bitrate, latency, frame rate, packet loss rate, and jitter from tests run across the wireless technologies 5G, 4G, and Wi-Fi 6. These results highlight key findings about performances achieved by the different wireless technologies. First, the bitrate remains relatively consistent across technologies, varying by less than five percent on average. Second, Wi-Fi 6 achieves the lowest latency, while 4G exhibits the highest—nearly an order of magnitude greater. Third, the frame rate observed at the system controller remains largely stable across all wireless technologies, with only a minor decrease noted for 5G. Fourth, the packet loss plot shows that both 4G and Wi-Fi 6 maintain losses below one percent, whereas 5G experiences approximately four percent. Finally, the 4G test demonstrates the highest average jitter at roughly one millisecond, while 5G and Wi-Fi 6 deliver more consistent timing performance.

indicating consistent encoder performance and adequate link capacity under the test conditions. The most notable differences arise in latency and jitter, where network design and scheduling mechanisms play a significant role. Wi-Fi 6 delivered the lowest latency and most consistent timing, suggesting that its higher bandwidth efficiency and reduced contention overheads are advantageous for real-time streaming at the edge. In contrast, 4G showed considerably higher latency and jitter, likely due to longer transmission scheduling intervals and core network delays inherent to LTE architectures. Although 5G achieved latency values lower than 4G, its elevated packet loss rate suggests intermittent instability—possibly related to network handovers or link adaptation at higher frequencies. Overall, the findings highlight that while all three technologies support real-time video streaming, Wi-Fi 6 offers the most reliable low-latency performance, 5G presents strong potential with room for optimization, and 4G remains the most constrained in terms of responsiveness and consistency.

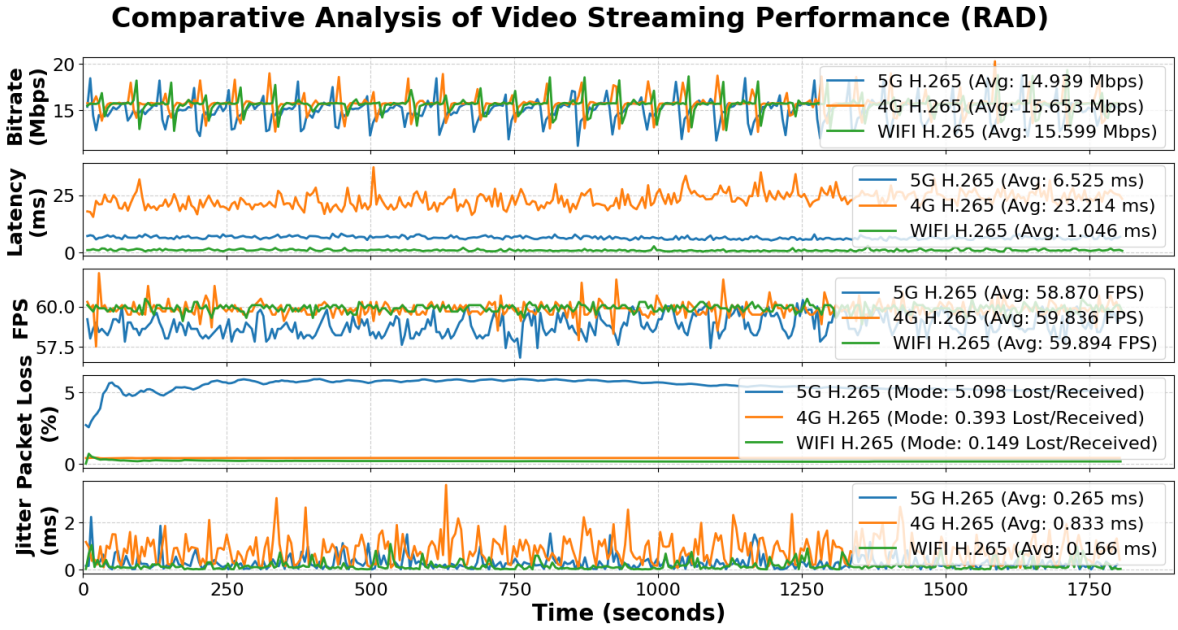

Figure 5.10: Raw Streaming Performance Data Across Wireless Technologies (Radxa/H.265). This figure displays a similar analysis as **Figure 5.9**, but the Radxa device was instead configured to use the H.265 encoder. These results reflect values nearly equal to those collected during the H.264 experiment; however, several differences can be observed. First, the average bitrate across the three wireless technologies is slightly lower, with the largest deviation not exceeding ten percent. Second, the ordering of latency remains consistent with that of the H.264 test, though both 4G and 5G exhibit modest increases by factors of 1.5 and 1.3, respectively. Third, the trend in packet loss rate persists, with 5G continuing to experience the highest loss, approximately 25% greater than in the H.264 test. Finally, jitter remains relatively low overall, though 4G demonstrates noticeably less stable delivery compared to 5G and Wi-Fi 6.

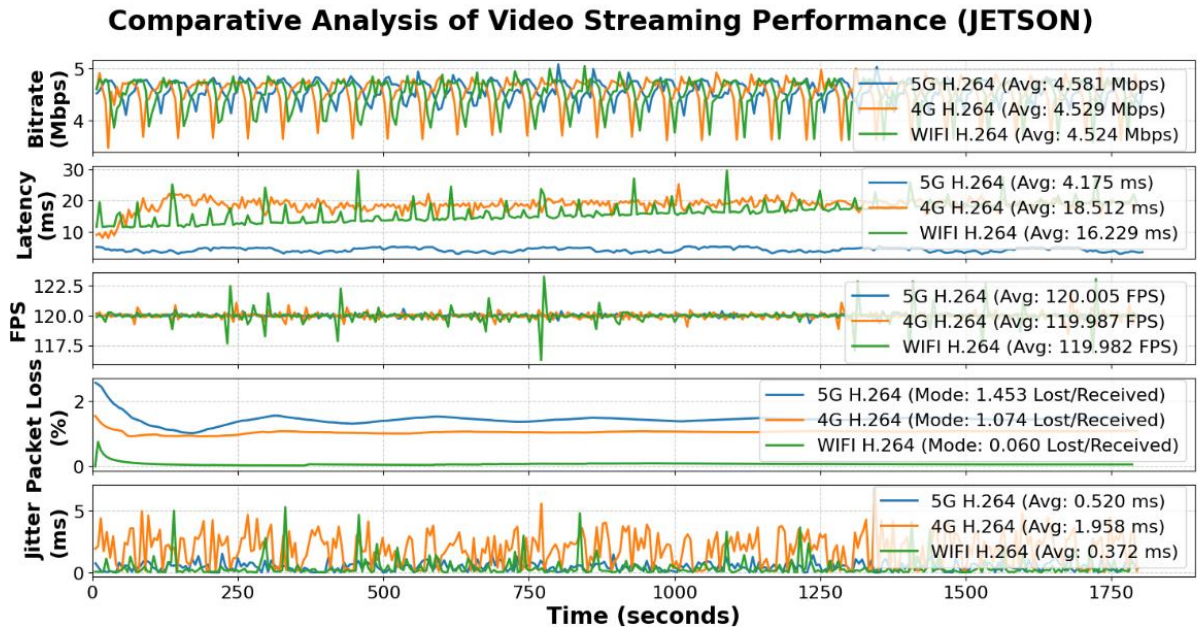

Figure 5.11: Raw Streaming Performance Data Across Wireless Technologies (Jetson/H.264). This figure displays a similar analysis as **Figure 5.9**, but it includes results from tests using the NVIDIA Jetson Orin NX device configured to use the H.264 encoder. The more powerful NVIDIA device achieved roughly double the frame rate of the Radxa device but did not exceed an average bitrate of 5 Mbps. A different trend was observed for latency, with 5G consistently exhibiting the lowest values. Wi-Fi 6, which ranked second, showed a gradual linear increase in latency over the course of the test. Packet loss rates remained relatively low overall, with 5G demonstrating a slightly higher rate and Wi-Fi 6 maintaining the lowest. Jitter increased across all three wireless technologies compared to the Radxa device, with 4G once again producing the greatest variation in delivery times.

The results in **Figure 5.10** show that network performance with the H.265 encoder closely mirrors that of the H.264 configuration, with only minor variations across the wireless technologies. Average bitrate decreased slightly, less than ten percent, while latency increased modestly for 4G and 5G. Packet loss followed a similar trend as before, with 5G exhibiting the highest rate, and jitter remained low overall, though 4G continued to show less stable delivery. These findings suggest that while the change in encoder influences throughput and timing slightly, the relative performance ranking among the wireless technologies remains consistent.

The small reduction in bitrate reflects H.265's greater compression efficiency, which reduces bandwidth demand without sacrificing output quality. The modest increases in latency are likely due to the additional computational overhead introduced by the H.265 codec's more complex encoding and decoding processes. Meanwhile, the higher packet loss observed on 5G may stem from link adaptation dynamics or transient instability in maintaining high-frequency

connections. The continued jitter advantage of Wi-Fi 6 and 5G over 4G can be attributed to their improved scheduling mechanisms and shorter transmission intervals, which promote more consistent packet delivery. Overall, these results highlight that encoder efficiency affects performance only marginally compared to the inherent characteristics of each wireless technology.

To build on the previous results and further isolate the influence of wireless technology on real-time video streaming performance, a second set of experiments was conducted using the NVIDIA Jetson Orin NX platform. Again, tests were conducted using the H.264 and H.265 encoders. These tests were not designed to compare hardware capabilities, but to provide additional evidence under similar network conditions, ensuring that observed trends could be attributed more directly to differences among the 4G, 5G, and Wi-Fi 6 links.

The results from the NVIDIA Jetson Orin NX experiments, ran using the H.264 encoder, reveal several notable differences compared to those obtained using the Radxa device, highlighting both the impact of the computing platform and the performance distinctions among

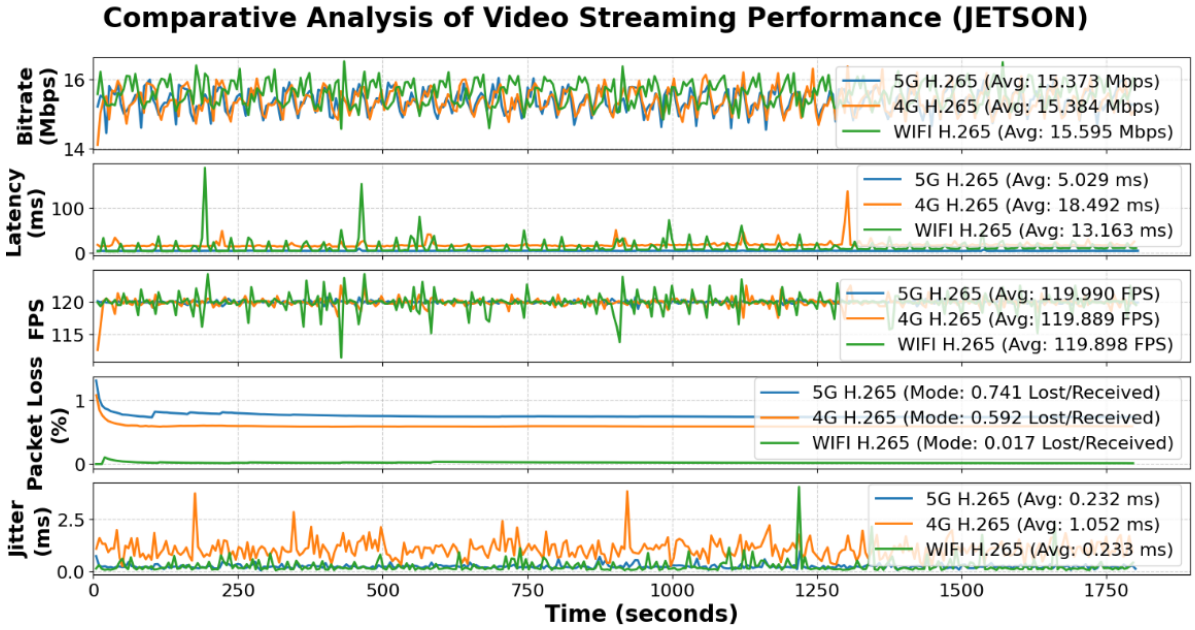

Figure 5.12: Raw Streaming Performance Data Across Wireless Technologies (Jetson/H.265). This figure displays a similar analysis as **Figure 5.9**, but it includes results from tests using the NVIDIA Jetson Orin NX device configured to use the H.265 encoder. The NVIDIA device's H.265 encoder produced more than three times the bitrate of the H.264 encoder at the same frame rate. As in previous tests, 5G yielded the lowest latency, while Wi-Fi 6 and 4G exhibited values more than twice as high. The packet loss rate remained consistent with that observed in the H.264 experiment. The 4G network again demonstrated the highest average jitter among the three technologies.

the wireless technologies. The most evident improvement is the substantial increase in frame rate, approximately double that of the Radxa system, indicating that the encoding process was less of a limiting factor in this configuration.

When comparing network performance, 5G demonstrated a clear advantage in latency, maintaining the lowest and most stable values throughout the test. Wi-Fi 6 followed closely but showed a gradual linear increase in latency, possibly reflecting congestion effects. Packet loss remained low across all wireless technologies, with Wi-Fi 6 achieving the most reliable delivery with 4G and 5G showing only a slight increase. Jitter values, however, rose for all three technologies relative to the Radxa trials, implying that higher frame rates and network scheduling interactions introduced greater short-term timing variability. Overall, these findings confirm that 5G offers superior responsiveness, Wi-Fi 6 delivers consistent throughput, and 4G continues to exhibit the greatest instability in packet timing.

The H.265 results on the NVIDIA Jetson Orin NX further reinforce and extend the performance distinctions observed among the wireless technologies while highlighting the influence of the more efficient codec. Compared to the H.264 tests, the H.265 encoder produced over three times the bitrate at an equivalent frame rate, indicating a substantial increase in compression efficiency and data throughput. Despite this higher transmission load, network behaviors remained consistent in their relative rankings—5G continued to deliver the lowest latency, while Wi-Fi 6 and 4G exhibited delays more than twice as high. This stability under increased bandwidth demand underscores the superior capacity and responsiveness of 5G for real-time streaming applications.

The packet loss rates showed minimal deviation from the earlier H.264 experiment, suggesting that all networks maintained reliable delivery even with the more data-intensive H.265 stream. However, jitter trends persisted, with 4G again showing the greatest variability in delivery timing, reflecting its sensitivity to fluctuating signal conditions and scheduling inefficiencies. In contrast, both 5G and Wi-Fi 6 maintained steadier timing despite the higher data rates, reinforcing their suitability for consistent low-latency streaming. Together, these results demonstrate that while encoder efficiency amplifies throughput demands, the relative performance hierarchy among wireless technologies remains unchanged—5G provides the best latency and reliability, Wi-Fi 6 offers balanced stability, and 4G continues to lag behind in real-time performance consistency.

Post Analysis Probability Distribution

The next set of figures provides a statistical perspective on the network performance results presented earlier by examining the probability density function (PDF) and cumulative distribution function (CDF) of the key performance metrics (KPM)—bitrate, latency, frame rate, packet loss, and jitter. These distributions offer a deeper understanding of how consistently each of the wireless technologies perform over time and how likely specific performance levels are to occur. The data were derived from the same experiments conducted on both the Radxa and NVIDIA platforms using the H.264 and H.265 encoders, ensuring that the analysis isolates the influence of network conditions rather than device or codec variability. **Figure 5.13** and **Figure 5.14** specifically illustrate the PDF and CDF of the received bitrate across 4G, 5G, and Wi-Fi 6 networks, providing insight into how each technology sustains throughput under comparable streaming workloads.

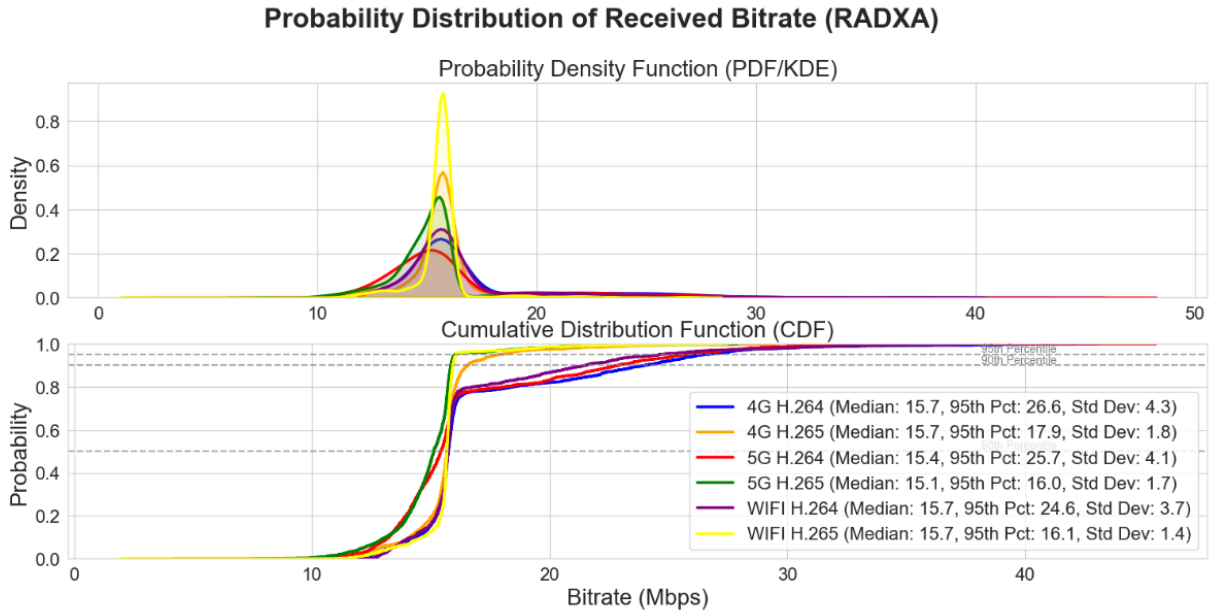

Figure 5.13: Probability Distribution of Received Bitrate Across Wireless Technologies (Radxa). This figure presents the probability density function (PDF) and cumulative distribution function (CDF) of the received bitrate from tests conducted on the Radxa Rock 5B device using the 4G, 5G, and Wi-Fi 6 wireless technologies. The results highlight an overall increase in bitrate accompanied by greater variation in the H.264 experiments across all three technologies, with the 95th percentile being approximately 1.6 times higher and the standard deviation nearly three times greater. Overall, the average bitrate among the three technologies differed only marginally.

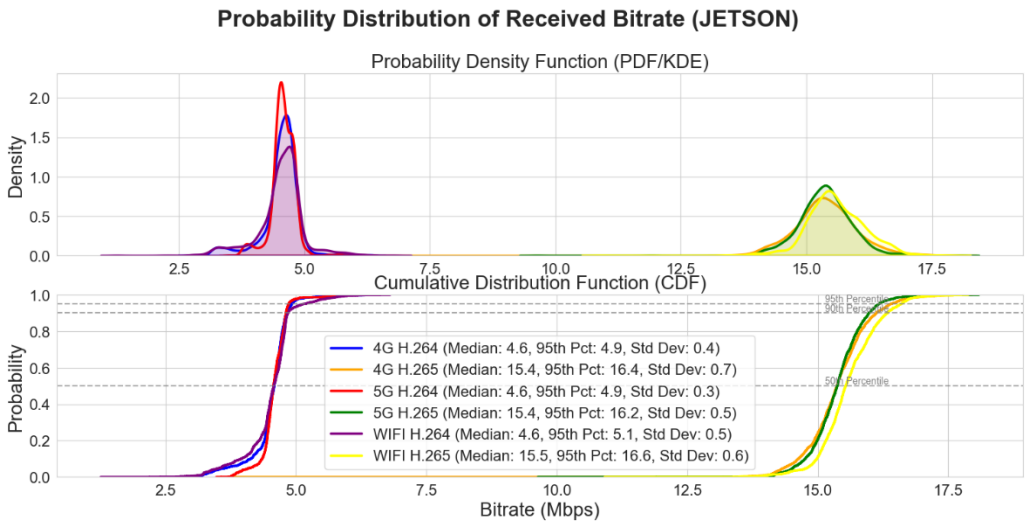

Figure 5.14: Probability Distribution of Received Bitrate Across Wireless Technologies (Jetson). This figure shows a similar analysis as **Figure 5.13**, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Unlike the results produced by the Radxa device, the 95th percentile for the H.265 encoder consistently reached values more than three times higher than those of the H.264 encoder. The distribution density around the median values in the H.264 tests was greater than that of the H.265 tests, with the largest difference observed for 4G, where the density was approximately 1.75 times higher.

The probability density and cumulative distribution analyses reveal clear differences in throughput characteristics between the Radxa Rock 5B and NVIDIA Jetson Orin NX platforms. For the Radxa device, the H.264 tests show a moderate increase in bitrate accompanied by substantial variation across all three wireless technologies, with the 95th percentile approximately 1.6 times higher than the median and a standard deviation nearly three times greater. Despite this variability, the average bitrate remained similar across 4G, 5G, and Wi-Fi 6, suggesting that the network, rather than the device, primarily constrained sustained throughput under the experimental conditions.

In contrast, the NVIDIA device demonstrates markedly higher throughput when using the H.265 encoder, with the 95th percentile exceeding three times that of the H.264 configuration at the same frame rate. The median density for H.264 tests was higher than for H.265, particularly for 4G where it was approximately 1.75 times greater, indicating that H.265 produces a wider spread of bitrate values around the median. These differences illustrate the combined effects of encoder efficiency and network capability: while the NVIDIA platform can sustain higher peak bitrates due to its processing capacity and H.265 compression, the relative performance of the wireless technologies remains evident. 5G consistently supports the highest throughput with the

lowest variability, Wi-Fi 6 offers moderate consistency, and 4G exhibits the most constrained and variable performance. Overall, the PDF and CDF analyses confirm that both the choice of encoder and underlying wireless technology strongly influence bitrate distribution and variability in real-time streaming scenarios.

Figure 5.15 and **Figure 5.16** show PDF and CDF plots of frame rate achieved by the system controller rendering frames to the applications display. These results will highlight the network's ability to consistently deliver complete frames.

The probability density and cumulative distribution analyses of frame rate reveal that both the Radxa Rock 5B and NVIDIA Jetson Orin NX platforms maintained highly consistent frame delivery across all three wireless technologies. For the Radxa device, frame rate differences among 4G, 5G, and Wi-Fi 6 were minimal, with a maximum variation of only two frames per second. The H.264 configuration exhibited slightly higher variability than H.265, with the standard deviation for 4G and 5G approximately 60% greater, indicating that the more computationally efficient H.265 encoder contributes to slightly more stable frame delivery.

The NVIDIA Orin NX results confirm this trend while highlighting the impact of device performance and network characteristics on frame rate consistency. Across all tests, the 95th percentile remained nearly identical, showing that peak frame rates were maintained regardless of encoder or wireless technology. Notably, 5G achieved the lowest standard deviation, at only 20% of the variability seen for Wi-Fi 6, suggesting that its scheduling mechanisms and lower latency contribute to more predictable frame delivery under higher throughput conditions. Overall, these findings indicate that while encoder efficiency and device capability influence minor variations, the choice of wireless technology, particularly 5G, plays a dominant role in achieving consistent, low-variance frame rates in real-time streaming scenarios.

The next plots, seen in **Figure 5.17** and **Figure 5.18**, show the PDF and CDF of one-way delay captured by the system controller. By examining these distributions, the figures provide insight into the consistency and variability of packet delivery across the three wireless technologies—4G, 5G, and Wi-Fi 6—under both H.264 and H.265 encoding conditions. Analyzing the latency distributions allows for a more detailed assessment of network performance beyond average values, highlighting the frequency of low-latency delivery, the presence of outliers, and the relative stability offered by each wireless technology. These figures are particularly relevant for understanding the suitability of each network for real-time video

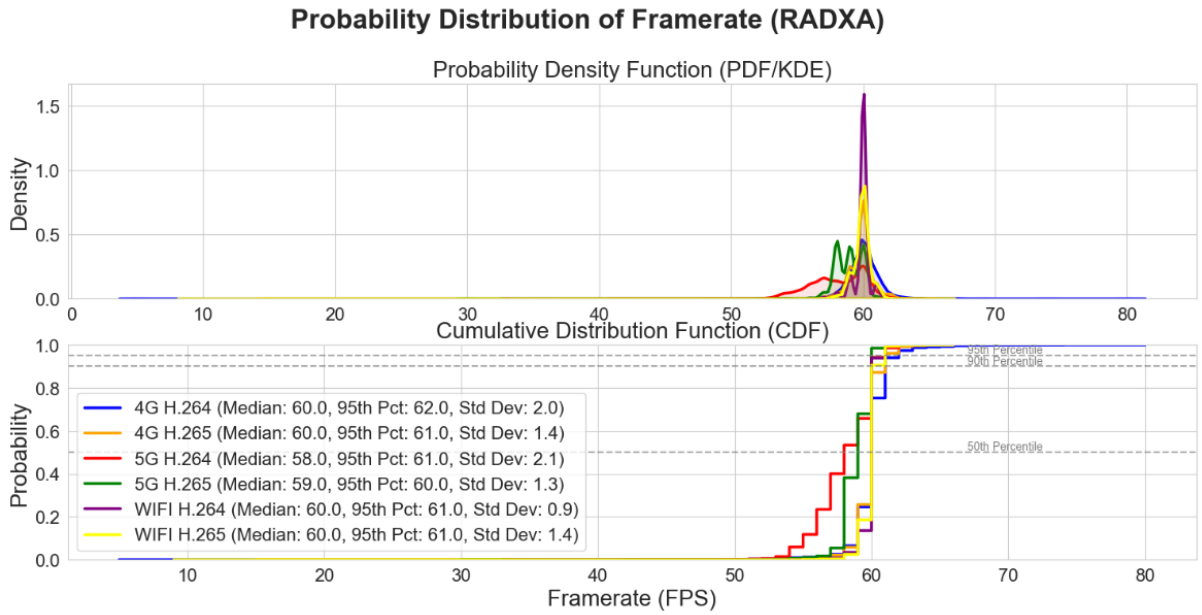

Figure 5.15: Probability Distribution of Framerate Across Wireless Technologies (Radxa). This figure shows the probability density function and cumulative distribution function for the framerate produced by tests ran with the Radxa Rock 5B device across the wireless technologies 4G, 5G, and Wi-Fi 6. The frame rate values across wireless technologies differed by no more than two frames per second. For the 4G and 5G tests, the standard deviation observed in the H.264 configuration was slightly higher, approximately 60% greater than that of the H.265 configuration.

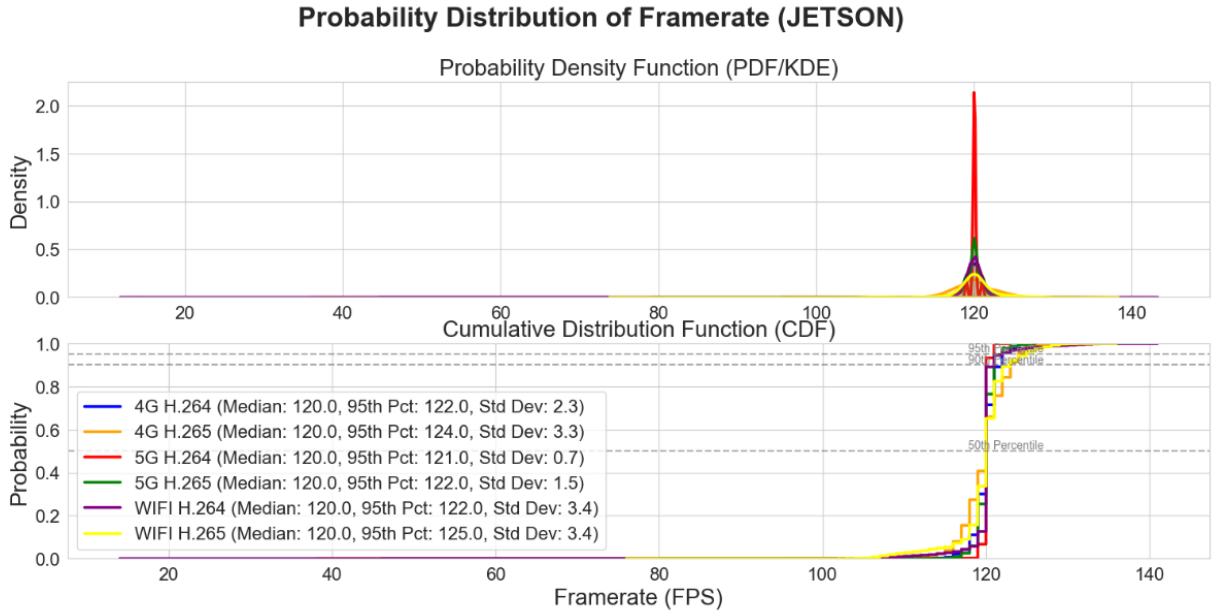

Figure 5.16: Probability Distribution of Framerate Across Wireless Technologies (Jetson). This figure shows a similar analysis as **Figure 5.15**, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Again, the 95th percentile remained almost equal across all tests. The lowest standard deviation was achieved by 5G at only twenty percent of the Wi-Fi 6 test.

streaming and for isolating the effects of network behavior from device processing capabilities.

The probability density and cumulative distribution analyses of end-to-end latency reveal

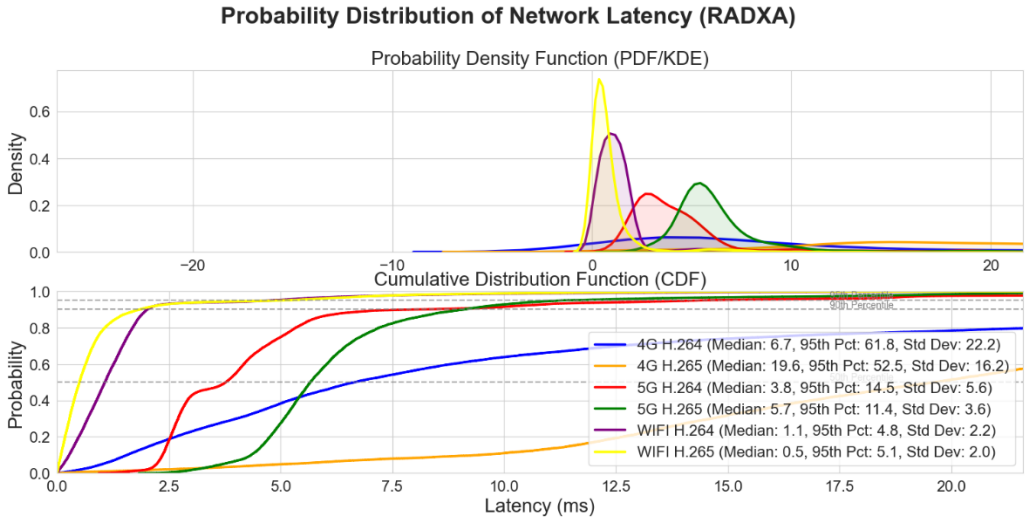

Figure 5.17: Probability Distribution of End-to-end Latency Across Wireless Technologies (Radxa). This figure shows the probability density function and cumulative distribution function for the end-to-end latency produced by tests ran with the Radxa Rock 5B device across the wireless technologies 4G, 5G, and Wi-Fi 6. Results varied considerably across the different configurations. Wi-Fi 6 demonstrated the best overall performance, with most latency values remaining around 5ms and a standard deviation of approximately 2ms. This performance is more than twelve times faster and ten times more stable than that of the slowest network, 4G.

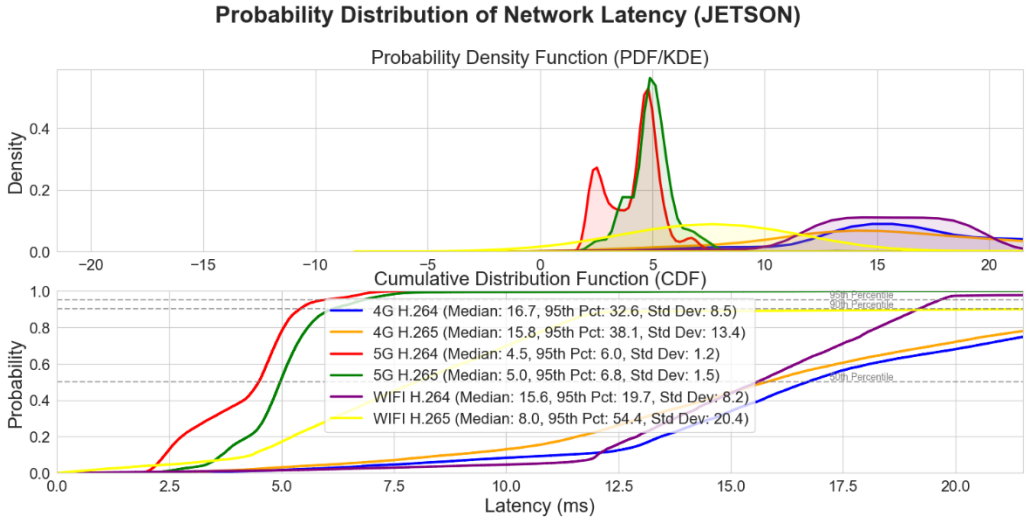

Figure 5.18: Probability Distribution of End-to-end Latency Across Wireless Technologies (Jetson). This figure shows a similar analysis as **Figure 5.17**, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Unlike the Radxa experiment, the NVIDIA device configuration experience the lowest latency and standard deviation with 5G enabled. The highest latency is observed in the Wi-Fi 6 results at eight times greater than 5G with a greater standard deviation by over thirteen times.

distinct differences in performance between the Radxa Rock 5B and NVIDIA Jetson Orin NX platforms and among the wireless technologies. For the Radxa device, Wi-Fi 6 demonstrated the lowest and most stable latency, with most values around 5 ms and a standard deviation of approximately 2 ms. In contrast, 4G exhibited the highest latency, more than twelve times slower than Wi-Fi 6, and with nearly ten times greater variability, highlighting the limitations of LTE scheduling and transmission intervals for real-time streaming.

The NVIDIA Orin NX experiments illustrate a shift in relative network performance under higher computational capacity. Here, 5G provided the lowest latency and tightest timing consistency, while Wi-Fi 6 exhibited the highest latency, approximately eight times greater than 5G, with a standard deviation exceeding 13 times that of 5G. These results suggest that under conditions of higher frame rates and increased processing throughput, 5G's low-latency design and optimized scheduling mechanisms are particularly effective, whereas Wi-Fi 6 can experience increased variability potentially due to contention or adaptive channel effects. Overall, the combined evidence underscores that while device capability influences absolute latency, the choice of wireless technology remains the dominant factor in achieving consistent, low-latency delivery for real-time streaming.

The PDF and CDF of the packet loss rate are shown in **Figure 5.19** and **Figure 5.20**. These statistics are paramount for determining the network's reliability and ability to facilitate real-time video streaming.

The probability density and cumulative distribution analyses of packet loss rate highlight clear differences in network behavior across wireless technologies and under different device workloads. For the Radxa Rock 5B, 5G exhibited the highest packet loss, reaching up to 6% with a variation of 1.3% throughout the experiment, while both 4G and Wi-Fi 6 maintained very low loss rates, with the 95th percentile below 0.5% and minimal variability. These results suggest that under moderate frame rates, 5G's higher sensitivity to transient link conditions or dynamic scheduling can lead to occasional packet drops, whereas 4G and Wi-Fi 6 provide more stable delivery.

Experiments with the NVIDIA Jetson Orin NX illustrate the impact of higher frame rates on packet loss behavior. With the increased processing load, 5G again experienced the greatest loss, though reduced to 1.5% under H.264, while 4G was a close second at 1%. Wi-Fi 6 continued to demonstrate nearly zero loss, confirming its reliability under both device

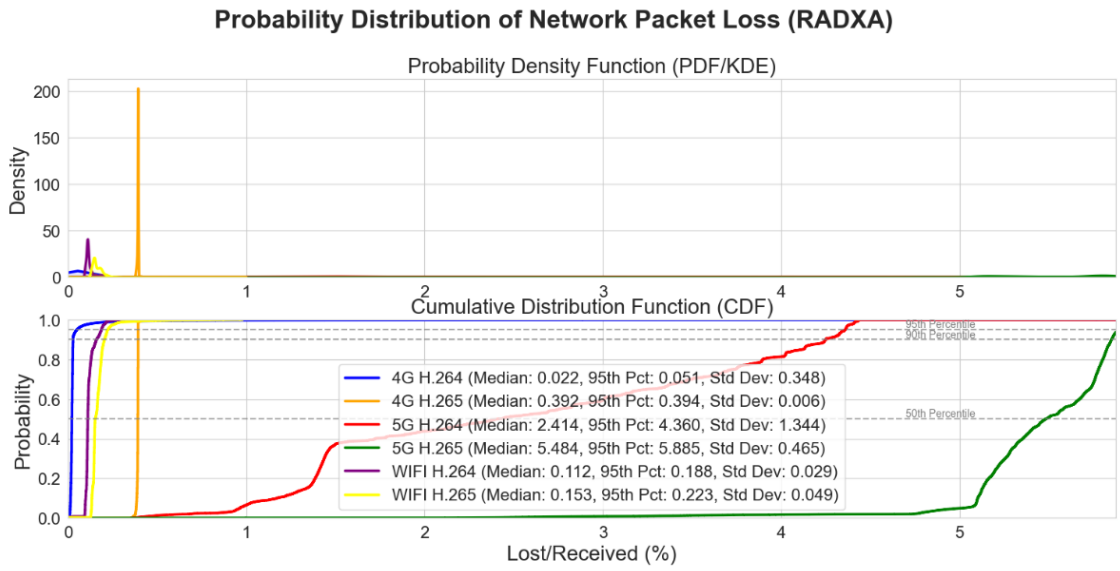

Figure 5.19: Probability Distribution of Packet Loss Rate Across Wireless Technologies (Radxa). This figure shows the probability density function and cumulative distribution function for the packet loss rate produced by tests ran with the Radxa Rock 5B device across the wireless technologies 4G, 5G, and Wi-Fi 6. The technology experiencing the greatest loss is 5G dropping at most 6% of packets with a variation reaching 1.3% throughout the experiment. Results for 4G and Wi-Fi 6 were relatively equal with a 95th percentile less than 0.5% and nearly zero variation.

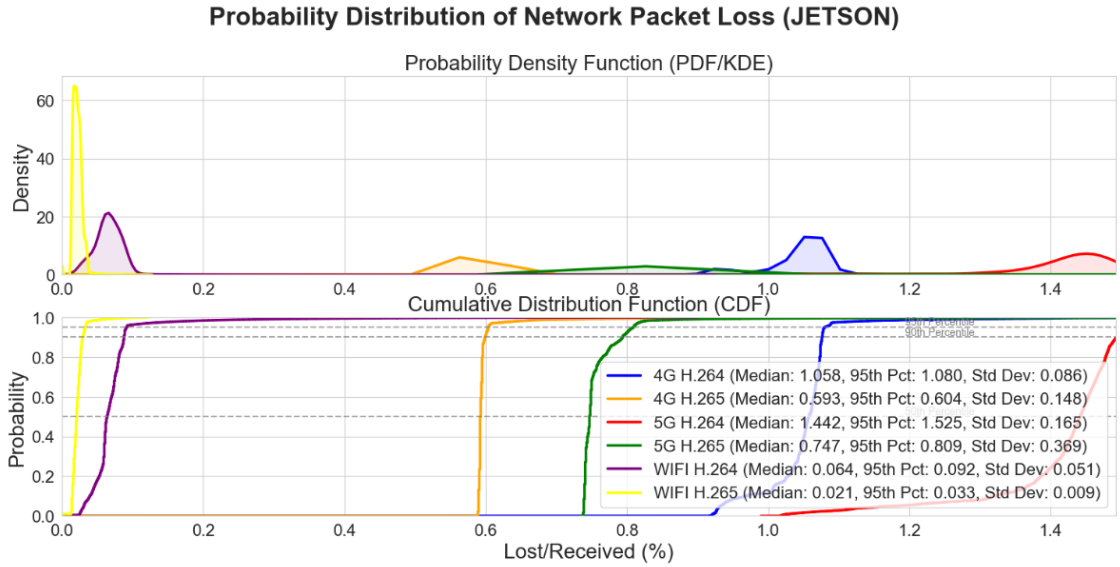

Figure 5.20: Probability Distribution of Packet Loss Rate Across Wireless Technologies (Jetson). This figure shows a similar analysis as **Figure 5.19**, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Results from this device with its higher framerate vary widely compared to the Radxa experiments. 5G again demonstrated the greatest loss, reaching 1.5% using H.264, and 4G was a close second reaching 1% using H.264. Wi-Fi 6 maintained a consistent loss rate at near zero.

configurations. The reduction in 5G packet loss compared to the Radxa experiments may reflect improved utilization of the network's higher throughput capacity or more effective link adaptation at the measured frame rates. Overall, these findings indicate that packet loss is strongly dependent on the wireless technology, with Wi-Fi 6 providing the most consistent delivery, while 5G is more variable and 4G remains intermediate, and that device throughput can modulate the absolute magnitude of losses without altering relative trends among the networks.

Figure 5.21 and **Figure 5.22** present the PDF and CDF of network jitter observed during the benchmarking experiments on both the Radxa Rock 5B and NVIDIA Jetson Orin NX devices. Jitter, defined as the variation in packet delivery times, is a critical metric for real-time video streaming, as excessive variability can lead to frame drops, uneven playback, and reduced quality of experience. By analyzing the statistical distribution of jitter across 4G, 5G, and Wi-Fi 6 networks, these figures provide insight into the consistency of packet delivery and the reliability of each wireless technology in supporting low-latency, smooth video transmission. The PDF and CDF representations allow a detailed assessment of both typical jitter values and the likelihood of extreme deviations, offering a comprehensive view of network stability under the tested conditions.

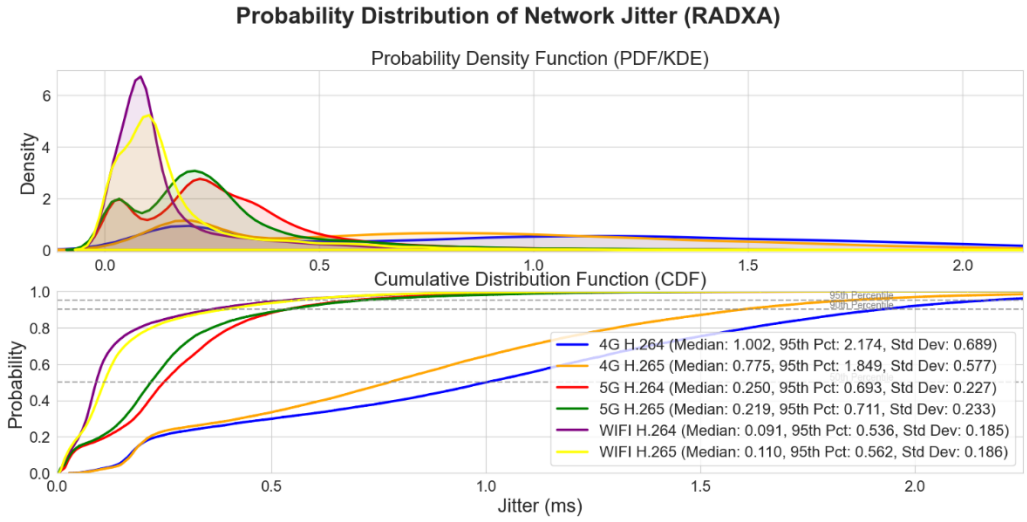

Figure 5.21: Probability Distribution of Average Jitter Across Wireless Technologies (Radxa). This figure shows the probability density function and cumulative distribution function for the average jitter produced by tests ran with the Radxa Rock 5B device across the wireless technologies 4G, 5G, and Wi-Fi 6. The greatest variation in packet delivery times was observed in the 4G configuration, reaching greater than 2ms, which also exhibited a substantially higher standard deviation by more than double that of Wi-Fi 6 and 5G. Wi-Fi maintained the lowest jitter below 0.5ms and low standard deviation around 0.2ms.

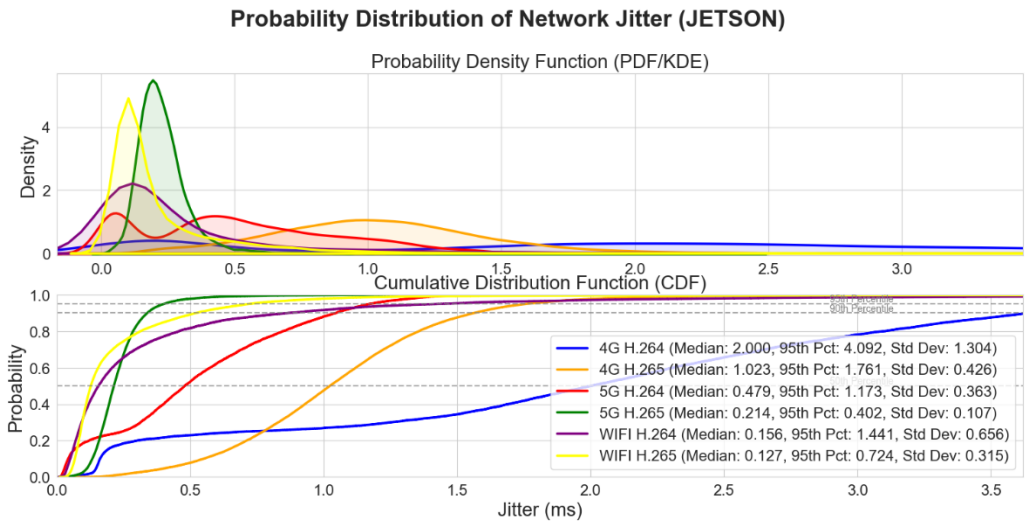

Figure 5.22: Probability Distribution of Average Jitter Across Wireless Technologies (Jetson). This figure shows a similar analysis as **Figure 5.21**, however it was produced by tests ran using the NVIDIA Jetson Orin NX. Like the Radxa results, 4G experienced the most jitter reaching values above 4ms with greater standard deviation. Wi-Fi 6 and 5G networks performed relatively equally, with H.264 results reaching above 1ms.

The probability density and cumulative distribution analyses of network jitter highlight clear differences in the stability of packet delivery across wireless technologies and devices. For the Radxa Rock 5B, 4G exhibited the greatest variation in packet delivery times, with jitter exceeding 2 ms and a standard deviation more than twice that of 5G and Wi-Fi 6. In contrast, Wi-Fi 6 maintained the lowest jitter, below 0.5 ms with a standard deviation around 0.2 ms, indicating highly consistent packet timing and minimal disruption for real-time streaming. 5G also showed low variability, though slightly higher than Wi-Fi 6.

The NVIDIA Jetson Orin NX results reinforce these trends while reflecting the impact of higher frame rates on jitter. Again, 4G experienced the largest jitter, exceeding 4 ms, with a correspondingly higher standard deviation. Wi-Fi 6 and 5G performed more similarly, with H.264 tests reaching jitter values above 1 ms, demonstrating increased variability relative to the Radxa device but maintaining relatively stable packet timing. These findings indicate that while increased processing and throughput can amplify absolute jitter values, the relative performance of the wireless technologies remains consistent: 4G is the most variable, 5G provides moderate stability, and Wi-Fi 6 offers the most reliable packet delivery rate for low-latency, real-time video streaming.

In summary, the benchmarking application effectively demonstrated how wireless technologies differ in their ability to support real-time video streaming at the edge. By

integrating both application- and network-level metrics, it captured the combined impact of link quality, processing load, and network type on end-to-end performance. The results showed Wi-Fi 6 providing the most stable and predictable performance, 5G offering competitive latency with moderate variability, and 4G exhibiting the highest jitter and delay. Across both hardware platforms, the observed trends were consistent, confirming that the benchmarking framework reliably characterizes network behavior under realistic streaming conditions and provides a robust means to evaluate communication technologies for edge-based video applications.

Performance Under Impairment

To evaluate the benchmarking application's ability to report real-time performance metrics under challenging conditions, a network impairment was introduced. Specifically, a surge of competing traffic was used to congest the link between the sensing node and the system controller. The impairment was activated once the application's metrics had reached a stable state, allowing observation of the network's dynamic response. During execution, screenshots of the GUI were captured for each key metric—bitrate, latency, jitter, frame rate, packet loss, and effective delivery rate—providing a visual record of real-time performance.

Figure 5.21 and **Figure 5.22** present the results from these experiments using the Radxa Rock 5B as the processing platform, with tests conducted on 4G and Wi-Fi 6 networks, respectively. Each plot represents the most recent trace of collected metrics, with the rightmost x-axis or time zero corresponding to the latest measurement. To highlight the moment of impairment, a vertical dashed gray line marks the time at which competing traffic was introduced. For readability, bitrate and frame rate plots display both per-second averages and global averages, while latency is presented alongside network jitter on a millisecond scale. Packet loss rate and effective delivery rate are plotted together on a percentage scale. Legends on each plot identify the corresponding metric, allowing clear interpretation of how the network responds to congestion. These visualizations are critical for understanding the resilience of each wireless technology and the system's ability to maintain smooth, low-latency video streaming under adverse conditions.

The real-time measurements captured during network impairments illustrate the sensitivity of video streaming performance to congestion across different wireless technologies. For the 4G experiment, the introduction of competing traffic caused a marked decline in both bitrate and frame rate, reflecting the network's reduced capacity to sustain high-throughput video

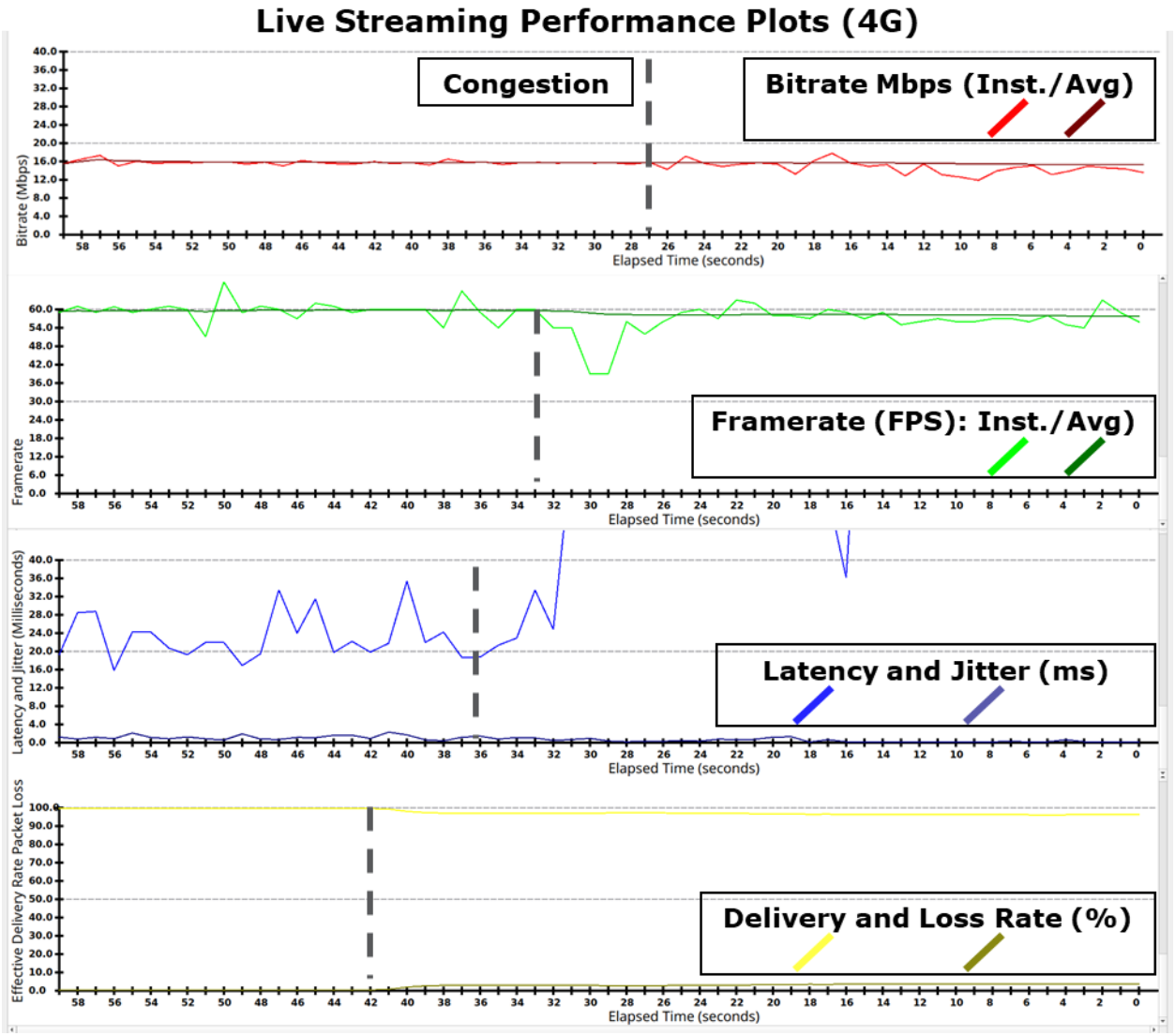

Figure 5.23: Screenshot of Application's Live Dynamic Plot During Traffic Congestion Impairment (4G/Radxa). The plots shown here are screenshots captured from the benchmarking application during operation over the 4G network under an induced network impairment. They display real-time measurements of bitrate, frame rate, latency, jitter, effective delivery rate, and packet loss rate while competing traffic was active during the test. The gray dashed lines mark the moment when network congestion was introduced. The results reveal a clear change in performance metrics following the onset of impairment: both bitrate and frame rate begin decreasing linearly, latency increases to more than twice its pre-impairment value, and packet loss rate rises slightly at the point of congestion activation.

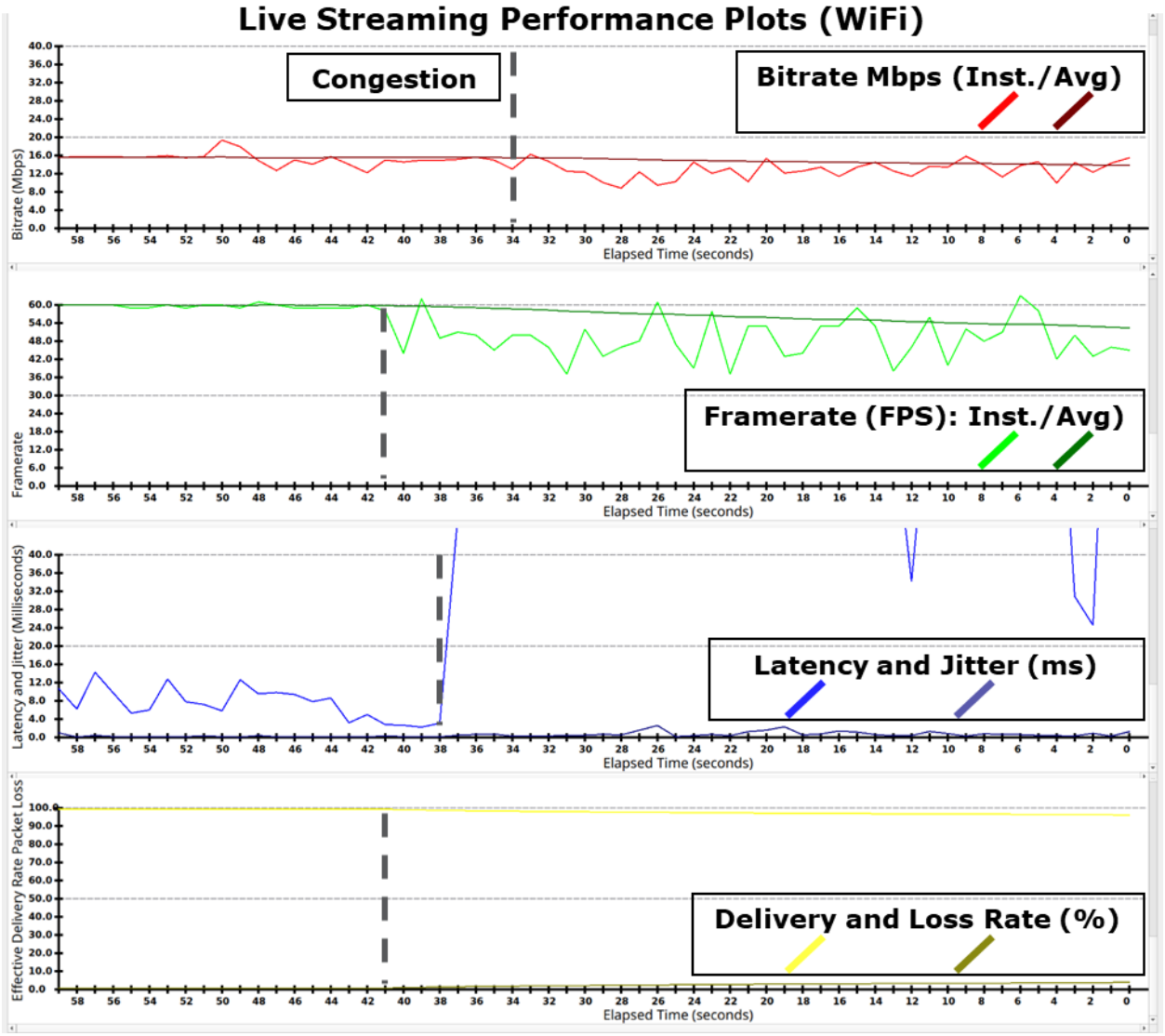

Figure 5.24: Screenshot of Application's Live Dynamic Plot During Traffic Congestion Impairment (WiFi/Radxa). This figure shows similar plots as **Figure 5.23**, but it was taken from an experiment with Wi-Fi 6 configured as the network. A similar trend is observed compared to the 4G test. Both bitrate and frame rate exhibit a linear decline, while latency rises sharply to more than four times its pre-impairment value. Packet loss rate also increases at the moment the impairment is activated.

delivery. Latency more than doubled relative to pre-impairment values, highlighting the impact of increased queuing and transmission delays, while packet loss rose slightly, indicating occasional dropped frames under load. Jitter and effective delivery rate trends, though less pronounced, also reflected the destabilizing effects of the impairment.

Similar behavior is observed in the Wi-Fi 6 experiment, though with amplified effects on latency. Following congestion, both bitrate and frame rate again declined linearly, while latency increased sharply to over four times its pre-impairment value. Packet loss similarly rose at the moment of activation, though Wi-Fi 6's higher baseline throughput mitigated some performance degradation relative to 4G. Overall, these results demonstrate that while both 4G and Wi-Fi 6 experience immediate and measurable impacts from network congestion, Wi-Fi 6 exhibits higher resilience in maintaining throughput, but latency remains highly sensitive to sudden surges in traffic. These findings underscore the importance of understanding dynamic network behavior for real-time video streaming, where even short-term congestion can significantly affect delivery quality.

In summary, the impairment experiments validated the benchmarking application's ability to capture real-time network dynamics and quantify the impact of congestion on streaming performance. By recording live metrics during induced traffic surges, the system effectively revealed how bitrate, latency, jitter, and frame rate evolve under load. The results confirmed that congestion produces immediate, measurable degradation across all key metrics, with Wi-Fi 6 showing greater throughput stability than 4G but similar sensitivity to latency spikes. Overall, the benchmarking tool proved effective for evaluating how wireless networks respond to transient impairments, providing a powerful means to assess their capacity to sustain reliable, low-latency video transmission in dynamic environments.

Discussion of Findings and Practical Implications

The results presented throughout this chapter demonstrate the critical interplay between hardware performance and wireless network behavior in determining the quality of real-time video streaming at the network edge. The benchmarking application developed in this study enabled precise measurement of application-level and network-level performance indicators, including bitrate, latency, jitter, and packet loss, under both normal and impaired conditions. By conducting controlled experiments using two heterogeneous processing platforms and multiple wireless technologies, the results offer valuable insights into how edge systems respond to

computational stress and dynamic network variability. The following subsections synthesize these findings to interpret their broader significance and implications for edge system design and deployment.

This Work vs. Established Frameworks / Standards

Benchmarking frameworks serve as the empirical foundation for evaluating the performance and reliability of edge computing and wireless communication systems. However, existing methodologies vary widely in their scope, realism, methods, and degree of synchronization between the network and application layers. This section examines several representative frameworks—OpenStack, EdgeBench, OpenRTiST, and the IEEE Standard for the Performance Assessment of Industrial Wireless Systems—in comparison with the unified benchmarking framework developed in this thesis. The goal is to highlight how each framework addresses key aspects such as workload realism, network modeling fidelity, synchronization precision, and cross-layer observability. By contrasting these approaches, the discussion identifies persistent gaps in reproducing real-world wireless edge conditions, particularly for mission-critical, real-time media workloads. The comparison summarized in **Table 5.1** illustrates how this thesis advances the state of benchmarking by integrating live, synchronized application and link-layer instrumentation across Wi-Fi 6, LTE, and 5G environments, thereby establishing a reproducible, standards-aligned method for evaluating performance under controlled yet realistic impairment scenarios.

Table 5.1: Comparison of Edge Benchmarking Frameworks.

Framework	Scope	Timing / Synchronization	Wireless Realism	Key limitation (relative to this thesis)	Major Contribution
OpenStack Telemetry	Cloud-oriented benchmarking and orchestration. Measures compute, storage, and networking perf. in virtualized environments.	event-driven; managed through system clocks and sampling intervals	Low - assumes virtualized data center networks	Does not capture joint app–radio behavior at the wireless edge.	Provides a comprehensive open-source framework for performance monitoring, resource allocation, and orchestration in large-scale cloud systems.

Table 5.2: Continued

Framework	Scope	Timing / Synchronization	Wireless Realism	Key limitation (relative to this thesis)	Major Contribution
EdgeBench	Benchmarking for edge computing primarily through workload latency, compute efficiency, and end-to-end inference delay in IoT and AI scenarios.	Synchronized timestamps between client and server; Microsecond-level timing through NTP	Moderate - Ethernet or controlled Wi-Fi environments with limited interference	Lacks synchronized link measurements and fine-grain media timing under real wireless contention.	Establishes reproducible methods for quantifying edge workload latency and energy efficiency across heterogeneous devices.
OpenRTiST	Real-time AI-assisted video streaming framework; demonstrates edge offloading and adaptive inference for interactive apps	Employs frame-level timestamps; synchronization achieved via RTP timebase	Moderate to High – utilizes real wireless links (Wi-Fi and LTE)	Excellent app realism, but less emphasis on synchronized channel telemetry and RAT-by-RAT comparability.	Demonstrates live edge inference and dynamic load distribution between client, edge, and cloud components for visual applications.
IEEE IWN perf. Standard	Defines standardized metrics, methodologies, and test procedures for evaluating the performance of industrial wireless networks and devices.	Does not define a specific implementation	Very High – specifically designed for industrial-grade wireless systems under realistic environmental and interference conditions.	Does not provide a realizable, live media pipeline or cross-layer logging system.	Establishes a formalized methodology and metric definitions for comparing industrial wireless technologies, serving as a benchmark reference for research and certification.
This Thesis	Unified benchmarking system for quantifying end-to-end latency, throughput, and reliability across Wi-Fi 6, 4G LTE, and 5G mmWave links in mission-critical edge applications.	High-precision timing achieved via embedded RTP timestamping, synchronized system clocks, and post-analysis of time-series metrics.	High – directly integrates commercial wireless modems, real signal propagation, and controlled impairments to capture realistic conditions	N/A	Provides a reproducible, real-time, multi-layer benchmarking framework linking physical, network, and application performance under real-world wireless environments.

While frameworks such as OpenStack provide valuable insight into the scalability and orchestration efficiency of cloud infrastructure, they inherently operate at a level of abstraction that omits the dynamics of wireless and real-time application behavior. Their focus on

virtualization overhead and provisioning latency offers a useful baseline for resource management studies, yet the workloads remain synthetic and detached from live media or edge-interactive contexts. Consequently, these frameworks cannot capture how fluctuations in the wireless channel, interference, or edge-node compute variability affect application-level performance. In contrast, the framework proposed in this thesis embeds the measurement process within a live, packetized media pipeline, allowing both the network and the application to be observed in real time under realistic wireless conditions.

In comparison, EdgeBench and OpenRTiST each contribute important perspectives on edge–cloud collaboration and real-time responsiveness. EdgeBench emphasizes measurable task latency for canonical IoT or AI workloads, providing meaningful end-to-end comparisons but with simplified or emulated network conditions. OpenRTiST, on the other hand, offers a more realistic, interactive workload by coupling live video with deep neural network inference, yet it lacks fine-grained synchronization and comprehensive radio-layer metrics. The thesis framework bridges these two paradigms by preserving the realism of OpenRTiST's live workload while incorporating the structured timing and telemetry rigor suggested by EdgeBench. Through per-frame one-way delay measurements embedded in RTP header extensions and synchronized logging of both link- and application-layer events, it delivers a more complete depiction of end-to-end system behavior across heterogeneous wireless technologies.

Finally, the IEEE Standard for the Performance Assessment of Industrial Wireless Systems provides a formal taxonomy of metrics and recommended synchronization methods but does not prescribe an executable or application-aware testbed. The framework developed in this thesis operationalizes that standard by implementing a standards-aligned measurement stack capable of live experimentation. By integrating controlled impairments—such as congestion, interference, and physical obstructions—with synchronized radio telemetry and application-level logging, the framework achieves both reproducibility and realism. This combination enables a cross-layer analysis that links radio conditions directly to perceptible application outcomes, a capability not addressed in prior frameworks. As a result, the unified real-time benchmark developed in this work offers a practical and extensible foundation for evaluating performance in mission-critical wireless edge deployments, closing the long-standing gap between controlled lab benchmarks and real-world operational environments.

Comparative Insights on Platform Performance

The comparative analysis between the Radxa Rock 5B and the NVIDIA Jetson Orin NX underscores the substantial influence of processing architecture on streaming performance. Both platforms exhibited predictable degradation in bitrate and framerate when computationally intensive tasks—such as facial detection—were enabled. However, the extent of degradation varied markedly between the two devices. The Radxa platform, equipped with a less capable CPU and limited hardware acceleration, demonstrated a pronounced drop in both bitrate and framerate, coupled with higher latency variability. In contrast, the Jetson Orin NX maintained greater stability and throughput under identical conditions, benefitting from its dedicated GPU cores, hardware encoders, and faster memory bandwidth.

These findings suggest that computational headroom is a key determinant of real-time streaming efficiency at the edge. When additional tasks such as machine learning inference are introduced, lower-powered devices may experience compounded latency as processing queues grow and encoding pipelines stall. Consequently, system designers must consider both the nominal performance of the hardware and its ability to sustain high throughput under load. The benchmarking framework effectively captured these differences, validating its utility as a diagnostic tool for characterizing the processing limits of heterogeneous edge platforms.

Network Behavior Under Load and Resilience to Impairment

Beyond hardware differences, the results under induced network congestion reveal the profound impact of wireless link characteristics on end-to-end video performance. The controlled introduction of competing traffic exposed distinct behavioral patterns across the 4G and Wi-Fi 6 networks. In both cases, the benchmarking application captured an immediate reduction in bitrate and framerate following the onset of impairment, accompanied by a measurable increase in latency and jitter. However, Wi-Fi 6 consistently demonstrated higher throughput stability and faster recovery after congestion, while 4G exhibited more severe latency inflation—rising to more than twice its steady-state value—and minor increases in packet loss.

These observations emphasize the temporal sensitivity of real-time streaming to transient congestion events. Even brief episodes of reduced bandwidth or increased queuing delay can significantly impair the smoothness and timeliness of video delivery. The differences between 4G and Wi-Fi 6 further highlight the role of underlying protocol design and medium access control in managing contention and delay. From a practical standpoint, these results underscore

the importance of selecting network technologies with predictable latency and low jitter characteristics when designing edge systems that rely on continuous media streaming or control feedback loops.

Implications for Edge System Design and Future Applications

Taken together, the experimental results provide both diagnostic insight and practical guidance for engineering reliable real-time video streaming at the edge. The data clearly shows that performance bottlenecks can arise from both computational and network constraints, and that the severity of these bottlenecks depends on how system load and link quality interact over time. To achieve predictable performance, designers must therefore adopt an integrated approach—balancing compute capabilities, encoding complexity, and wireless link characteristics to maintain consistent throughput and low latency.

The benchmarking application itself proved to be a valuable framework for this purpose, offering fine-grained visibility into how metrics evolve dynamically under varying workloads and network conditions. By combining real-time visualization with automated data logging, the tool demonstrated its capacity to identify thresholds of stability and reveal subtle degradations before they become critical. These attributes make it particularly well suited for evaluating next-generation wireless systems such as 5G and beyond, where adaptive rate control, edge inference, and cross-layer optimization are becoming central to application performance.

In summary, the findings confirm that the benchmarking methodology developed in this study provides an effective means of linking measurable performance outcomes to underlying system behaviors. Its ability to isolate and quantify the effects of hardware and network conditions makes it a practical foundation for future research in adaptive streaming, edge intelligence, and resilient wireless communication systems.

Conclusion

This chapter translated synchronized experimental data into a coherent understanding of how compute capacity, codec configuration, and wireless technology jointly shape the performance of real-time streaming at the edge. Through controlled and repeatable trials, the results demonstrated that device processing headroom fundamentally constrains frame stability and throughput, while the chosen RAT dictates latency, jitter, and loss behavior across operating conditions. By aligning application-level metrics with channel telemetry, the analysis provided a causal view of performance—showing not only which configurations perform best, but why.

The findings confirmed that computational and network effects are intertwined: insufficient processing resources propagate directly into the media stream as unstable pacing and delayed transmission, while network variability can amplify or mask those effects depending on the underlying RAT. Wi-Fi 6 exhibited the most predictable low-latency behavior, 5G offered a balance of speed and flexibility, and 4G revealed the highest timing variability—evidence of the hierarchical delay and jitter regimes inherent to each technology. Controlled impairment tests further validated the framework's sensitivity, capturing transient degradations that mirrored real-world congestion events and revealing how system resilience differs across network types.

Together, these results verify the methodological soundness of the benchmarking framework. The consistent patterns observed across devices, codecs, and RATs demonstrate that the experimental design successfully isolates the impact of each factor while preserving realism. Moreover, the alignment between raw traces, statistical distributions, and live visualizations confirms that the framework measures both steady-state and transient performance with diagnostic fidelity suitable for mission-critical edge applications.

The insights distilled from this analysis offer practical design guidance: match the wireless interface to timing requirements, maintain adequate compute headroom for stable media pacing, treat codec selection as an adaptive parameter rather than a constant, and anticipate transient load effects through synchronized monitoring. While the quantitative outcomes are specific to the tested hardware and configurations, the qualitative relationships they reveal are broadly applicable to edge systems operating under tight latency constraints.

Having established empirical baselines and verified the framework's analytical depth, Chapter Six integrates these results into a broader synthesis—discussing their implications for cross-layer system design, adaptive control strategies, and future extensions to more diverse workloads and wireless environments.

CHAPTER SIX
CONCLUSION

Summary

This thesis presented the design, implementation, and evaluation of a unified benchmarking framework for assessing the performance of edge computing systems deployed over modern wireless communication technologies. The framework was developed to quantify the impact of wireless network characteristics and processing constraints on real-time, mission-critical applications. Its overarching objective was to provide a practical, reproducible, and extensible approach to measuring the real-world capabilities of consumer-grade edge platforms when subjected to realistic operating conditions across multiple network domains, including Wi-Fi 6, 4G LTE, and 5G mmWave.

The study began by identifying the growing importance of edge computing in domains where responsiveness, reliability, and data integrity are paramount. These include critical infrastructure systems such as transportation, automation, and public safety, where computational tasks must be executed close to the data source to meet strict latency and availability requirements. The literature review established the foundational concepts underpinning the research, surveying edge computing paradigms, Multi-access Edge Computing (MEC) architectures, and the evolution of modern wireless standards. It also examined prior benchmarking frameworks and their limitations, emphasizing the need for a unified, real-world methodology capable of evaluating both application-layer behavior and underlying network conditions in a synchronized manner.

Building upon this context, the thesis detailed the design of the unified benchmarking framework, which integrates three primary components: the sensing node, the network gateway, and the system controller. The sensing node emulates a mission-critical endpoint that performs real-time data capture, processing, and transmission. The network gateway represents the wireless communication medium, enabling controlled experimentation across different network technologies and conditions. The system controller functions as the central monitoring and visualization platform, providing real-time feedback and data logging. Together, these components form a closed-loop benchmarking environment capable of both live and post-analysis of performance metrics.

The implementation of the framework leveraged open-source technologies, primarily the GStreamer multimedia pipeline, which was customized for real-time measurement and analysis. The streaming subsystem utilized the Real-time Transport Protocol (RTP), enhanced with timestamp embedding and extraction mechanisms to facilitate precise end-to-end latency measurement. Video data served as the benchmark workload, simulating a realistic edge AI application through facial feature detection and recognition. This workload introduced measurable processing overhead, allowing for the simultaneous evaluation of computational and communication performance. Performance metrics such as bitrate, frame rate, latency, packet loss, and jitter were continuously captured during each experimental trial.

Experimental evaluation was conducted using multiple processing platforms—specifically, the NVIDIA Jetson Orin NX and the Radxa Rock 5B—operating across various wireless configurations. The experimental methodology was rigorously structured, with controlled baselines established using loopback interfaces to isolate computational performance from network effects. Subsequent trials introduced Wi-Fi 6, 4G LTE, and 5G mmWave links, followed by impairment tests designed to emulate congestion. Each experimental scenario was repeated with and without on-device processing to capture the impact of computational workloads on network throughput and latency. The resulting data was analyzed both in real time and through post-processing of time-series distributions, providing a comprehensive statistical view of performance across layers.

The results demonstrated clear dependencies between processing capabilities, network characteristics, and application behavior. Wi-Fi 6 and 5G consistently achieved lower latency and higher stability than 4G LTE, while the Jetson Orin NX outperformed lower-tier platforms in maintaining frame rate and bitrate under heavy computational load. Probability density analyses revealed that 5G's millimeter-wave channel, while capable of high throughput, exhibited susceptibility to increased variance under minor signal degradation. Conversely, Wi-Fi 6 maintained steady throughput with minimal jitter, highlighting its robustness for localized edge deployments. The inclusion of network impairments confirmed predictable degradation trends, providing empirical validation for the benchmarking framework's ability to capture real-world performance dynamics.

Through extensive data collection and interpretation, this research successfully established the relationships between wireless link conditions, edge device performance, and the

quality of experience delivered to mission-critical applications. The findings underscore the feasibility of using compact, cost-effective hardware to support latency-sensitive workloads at the edge, provided that system design accounts for the inherent variability of wireless environments. Collectively, the framework, methodology, and empirical results provide a valuable foundation for future research into the deployment and optimization of distributed edge systems operating under real-world conditions.

Contributions

The principal outcome of this work is the creation of a unified, real-time benchmarking framework for evaluating edge computing performance across heterogeneous wireless networks. This framework bridges a critical methodological gap in the field by integrating synchronized, multi-layer data acquisition and analysis, thereby enabling comprehensive assessment of edge system performance under realistic operational constraints. Its modular architecture allows for reproducible experimentation across diverse hardware platforms and network configurations, contributing a scalable and extensible foundation for future benchmarking research.

Major Contribution 1: Unified Benchmarking Framework Design.

The first and foremost contribution lies in the design and implementation of a flexible benchmarking architecture that encompasses sensing, processing, and communication components. Unlike prior approaches that rely on simulated workloads or idealized conditions, this framework operates using real-time video streams and hardware-based measurements, ensuring that results faithfully represent actual edge deployment scenarios. The integration of physical-layer sensing, application-level monitoring, and synchronized metric logging provides a comprehensive view of system performance. This capability enables direct comparison of heterogeneous configurations and fosters empirical validation of theoretical models of wireless and computational performance.

Major Contribution 2: Quantitative Characterization of Edge System Behavior.

The second major contribution is the rigorous quantification of application performance in dynamic network environments. By intentionally introducing controlled impairments such as bandwidth reduction, latency injection, and congestion, the study demonstrated predictable, measurable impacts on critical metrics including bitrate, delay, and loss. These controlled experiments validated the framework's sensitivity to network variation and its utility for evaluating resilience strategies. The empirical dataset produced through these trials provides an

unprecedented quantitative basis for understanding the interaction between wireless link quality and application responsiveness in mission-critical contexts.

Minor Contribution: Protocol and Software Optimization.

A secondary but significant contribution is the refinement of streaming protocols and compression techniques implemented within the framework. Custom modifications to the GStreamer pipelines, particularly the embedding of timestamps in RTP header extensions, allowed for precise latency measurement. Additionally, optimization of codec selection and configuration, particularly in H.264 and H.265 streaming modes, enhanced the realism and generalizability of the experiments. These refinements not only improved system efficiency but also demonstrated best practices for optimizing resource-limited edge platforms. Collectively, these innovations advance the state of practical benchmarking and establish the framework as a reference model for evaluating edge systems in both academic and industrial settings.

The combination of these contributions positions this work as both a methodological and practical advancement in the field of edge performance evaluation. It provides a reproducible toolchain, validated experimental results, and actionable insights that collectively enable more informed design of future wireless and edge-integrated systems.

Future work

While this research achieved its objectives in constructing and validating a unified benchmarking framework, several avenues remain open for expansion and refinement. The modularity of the current system allows for seamless integration of new components, environments, and workloads, presenting opportunities to broaden both the scope and fidelity of future studies.

Integration with Advanced Network Techniques

A promising direction for future research involves extending the benchmarking framework to support advanced network paradigms such as Software-Defined Networking (SDN), network slicing, and emerging 6G architectures. SDN offers centralized, programmable control over network traffic flows, enabling dynamic reconfiguration and policy-based optimization that can directly impact latency, throughput, and reliability. Incorporating SDN into the experimental setup would allow for precise orchestration of network resources and fine-grained performance tuning during real-time workloads. Similarly, network slicing—a core concept in 5G and beyond—can be leveraged to isolate benchmarking traffic into customized

logical networks with specific quality-of-service (QoS) guarantees, facilitating comparative analysis of resource allocation strategies. Looking further ahead, the advent of 6G networks promises to integrate artificial intelligence-driven orchestration, terahertz communication, and semantic networking. Adapting the framework for compatibility with early 6G testbeds will enable exploration of ultra-low-latency, high-capacity links, and the co-design of intelligent edge services. Integrating these emerging technologies will position the benchmarking framework as a critical tool for validating performance in next-generation, software-defined and AI-optimized network environments.

Integration of Diverse Sensing Node Types.

Future iterations of the benchmarking framework should incorporate additional categories of sensing nodes, such as mini-computers, FPGAs, and embedded microcontroller-based systems. Each represents a distinct balance between computational capacity, energy efficiency, and I/O performance. Evaluating these devices will extend the framework's applicability to a broader range of industrial and IoT use cases, from low-power environmental sensors to high-throughput, hardware-accelerated video processors. FPGA-based nodes, in particular, offer potential for deterministic processing and low-latency networking, which could be benchmarked against conventional CPU- and GPU-based platforms to assess suitability for ultra-reliable low-latency communication (URLLC) scenarios.

Maximizing Bandwidth and Network Capacity Utilization.

Another area for future investigation involves pushing the limits of available network bandwidth and channel capacity. This may include experiments designed to saturate wireless links using multi-stream or parallel transmission configurations, allowing researchers to observe the onset of congestion, packet loss, and throughput collapse. Such experiments could reveal the operational thresholds of emerging wireless standards and inform adaptive bitrate or congestion control mechanisms suitable for edge applications that must dynamically adjust to fluctuating channel conditions.

Scaling to Dense, Distributed Sensing Networks.

Expanding the framework to support a larger number of simultaneous sensing nodes would enable the study of distributed edge systems operating at scale. This would allow exploration of resource contention, coordination latency, and collective performance behavior in dense deployments such as industrial facilities or urban sensor networks. The implementation of

distributed synchronization, time alignment across multiple nodes, and centralized aggregation of performance metrics would represent significant extensions of the framework's current single-node architecture.

Broader Multimedia Workloads and Video Source Diversity.

The present work utilized a fixed video source and resolution to maintain experimental consistency. Future experiments should vary these parameters to examine how different input characteristics—such as frame rate, resolution, compression level, and scene complexity—affect end-to-end performance. Such studies would allow for more comprehensive modeling of computational load and network utilization, providing practical guidance for designing adaptive encoding and transmission strategies tailored to application requirements and available bandwidth.

Evaluation of Alternative Streaming Protocols.

Future research should also explore the implementation of alternative streaming protocols such as Dynamic Adaptive Streaming over HTTP (DASH), MPEG transport streams, and Web Real-Time Communication (WebRTC). Each of these protocols offers distinct mechanisms for handling congestion, packet loss, and synchronization. Comparing their behavior under identical network conditions would extend the framework's utility for application developers seeking optimal trade-offs between latency, quality, and robustness across diverse edge scenarios.

Deployment in Outdoor and Complex Network Environments.

Finally, expanding testing environments beyond controlled laboratory conditions will be essential for evaluating system performance under realistic, variable conditions. Outdoor and complex environments—characterized by mobility, multipath interference, and variable atmospheric conditions—pose challenges not encountered in static setups. Conducting experiments in such environments would provide valuable insights into the resilience of wireless edge systems and further validate the framework's ability to characterize performance across physical and environmental extremes.

LIST OF REFERENCES

[1] M. Satyanarayanan, "The Emergence of Edge Computing," *Computer,* vol. 50, no. 1, pp. 30–39, 2017, doi: 10.1109/mc.2017.9.

[2] J. Ren, D. Zhang, S. He, Y. Zhang, and T. Li, "A Survey on End-Edge-Cloud Orchestrated Network Computing Paradigms," *ACM Computing Surveys,* vol. 52, no. 6, pp. 1–36, 2019, doi: 10.1145/3362031.

[3] S. T. Spantideas, A. E. Giannopoulos, and P. Trakadas, "Smart Mission Critical Service Management: Architecture, Deployment Options, and Experimental Results," *IEEE Transactions on Network and Service Management,* vol. 22, no. 2, pp. 1108–1128, 2025, doi: 10.1109/tnsm.2024.3498348.

[4] H. A. Alharbi and M. Aldossary, "Energy-Efficient Edge-Fog-Cloud Architecture for IoT-Based Smart Agriculture Environment," *IEEE Access,* vol. 9, pp. 110480–110492, 2021, doi: 10.1109/access.2021.3101397.

[5] Z. Hamidi-Alaoui and A. E. Elalaoui, "A WSN-based approach for prioritising emergency vehicles : Performance analysis of medium access control," 2018 2018: IEEE, doi: 10.1109/icoa.2018.8370581. [Online]. Available: https://dx.doi.org/10.1109/icoa.2018.8370581

[6] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge Computing: Vision and Challenges," *IEEE Internet of Things Journal,* vol. 3, no. 5, pp. 637–646, 2016, doi: 10.1109/jiot.2016.2579198.

[7] P. Arthurs, L. Gillam, P. Krause, N. Wang, K. Halder, and A. Mouzakitis, "A Taxonomy and Survey of Edge Cloud Computing for Intelligent Transportation Systems and Connected Vehicles," *IEEE Transactions on Intelligent Transportation Systems,* vol. 23, no. 7, pp. 6206–6221, 2022, doi: 10.1109/tits.2021.3084396.

[8] O. Friha, M. A. Ferrag, L. Shu, L. Maglaras, and X. Wang, "Internet of Things for the Future of Smart Agriculture: A Comprehensive Survey of Emerging Technologies," *IEEE/CAA Journal of Automatica Sinica,* vol. 8, no. 4, pp. 718–752, 2021, doi: 10.1109/jas.2021.1003925.

[9] S. Baidya, Y.-J. Ku, H. Zhao, J. Zhao, and S. Dey, "Vehicular and Edge Computing for Emerging Connected and Autonomous Vehicle Applications," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020 2020: IEEE, doi:

10.1109/dac18072.2020.9218618. [Online]. Available: https://dx.doi.org/10.1109/dac18072.2020.9218618

[10] A. Jain and D. S. Jat, "An Edge Computing Paradigm for Time-Sensitive Applications," 2020 2020: IEEE, doi: 10.1109/worlds450073.2020.9210325. [Online]. Available: https://dx.doi.org/10.1109/worlds450073.2020.9210325

[11] S. Homayouni *et al.*, "On the Feasibility of Cellular-Connected Drones in Existing 4G/5G Networks: Field Trials," in *2021 IEEE 4th 5G World Forum (5GWF)*, 2021 2021: IEEE, doi: 10.1109/5gwf52925.2021.00057. [Online]. Available: https://dx.doi.org/10.1109/5gwf52925.2021.00057

[12] S. Waisara, S. Charoenlarpnopparut, C. Srisurangkul, and T. Nishio, "Vehicle Telematics System Design for Real-time Applications using Mobile Networks," in *2023 18th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)*, 2023 2023: IEEE, pp. 1–6, doi: 10.1109/isai-nlp60301.2023.10354548. [Online]. Available: https://dx.doi.org/10.1109/isai-nlp60301.2023.10354548

[13] M. Rady, O. Iova, H. Rivano, A. Deligianni, and L. Drikos, "How does Wi-Fi 6 fare? An industrial outdoor robotic scenario," *Ad hoc networks.,* vol. 156, p. 103418, 2024, doi: 10.1016/j.adhoc.2024.103418.

[14] E. Ramadan, A. Narayanan, U. K. Dayalan, R. A. K. Fezeu, F. Qian, and Z.-L. Zhang, "Case for 5G-aware video streaming applications," 2021 2021: ACM, pp. 27–34, doi: 10.1145/3472771.3474036. [Online]. Available: https://dx.doi.org/10.1145/3472771.3474036

[15] I. Khan, T. X. Tran, M. Hiltunen, T. Karagioules, and D. Koutsonikolas, "An Experimental Study of Low-Latency Video Streaming over 5G," 2024, doi: 10.48550/arxiv.2403.00752.

[16] B. Varghese *et al.*, "A Survey on Edge Performance Benchmarking," *ACM Computing Surveys,* vol. 54, no. 3, pp. 1–33, 2022, doi: 10.1145/3444692.

[17] A. Das, S. Patterson, and M. P. Wittie, "EdgeBench: Benchmarking Edge Computing Platforms," 2018, doi: 10.48550/arxiv.1811.05948.

[18] K. A. Kumari, G. S. Sadasivam, D. Dharani, and M. Niranjanamurthy, *Edge computing : fundamentals, advances and applications*, First edition. ed. Boca Raton, FL: CRC Press, 2022.

[19] D. I. Inc. "New York City DOT Deploys Digi Solutions to 14k Intersections with Digi Remote Manager." https://www.digi.com/resources/customer-stories/new-york-city-dot-deploys-digi-solutions (accessed 2025).

[20] S. Mazur. "NYC Intelligent Transportation Project Wins ITS-NY Award, Advancing ITS." Digi International Inc. https://www.digi.com/blog/post/nyc-intelligent-transportation-project-wins-its-ny (accessed 2025).

[21] M. Iorio, F. Risso, and C. Casetti, "When Latency Matters: Measurements and Lessons Learned," *ACM SIGCOMM Computer Communication Review,* vol. 51, no. 4, pp. 2–13, 2021, doi: 10.1145/3503954.3503956.

[22] Z. Kaleem *et al.*, "UAV-Empowered Disaster-Resilient Edge Architecture for Delay-Sensitive Communication," *IEEE Network,* vol. 33, no. 6, pp. 124–132, 2019, doi: 10.1109/mnet.2019.1800431.

[23] Y. Zhang, *Mobile Edge Computing* (Simula SpringerBriefs on Computing ; v.9). Cham: Springer International Publishing AG, 2021.

[24] ETSI, "Multi-access Edge Computing (MEC); Framework and Reference Architecture," ETSI GS MEC 003 V4.1.1, 2025.

[25] R. Zhang *et al.*, "Joint Task Offloading and Resource Allocation in UAV-Assisted MEC Networks for Disaster Rescue: A Large AI Model Enabled DRL Approach," *IEEE Internet of Things Journal,* pp. 1–1, 2025, doi: 10.1109/jiot.2025.3605692.

[26] M. K. Mondal, S. Banerjee, D. Das, U. Ghosh, M. S. Al-Numay, and U. Biswas, "Toward Energy-Efficient and Cost-Effective Task Offloading in Mobile Edge Computing for Intelligent Surveillance Systems," *IEEE Transactions on Consumer Electronics,* vol. 70, no. 1, pp. 4087–4094, 2024, doi: 10.1109/tce.2024.3362396.

[27] A. Seferagić, J. Famaey, E. De Poorter, and J. Hoebeke, "Survey on Wireless Technology Trade-Offs for the Industrial Internet of Things," *Sensors,* vol. 20, no. 2, p. 488, 2020, doi: 10.3390/s20020488.

[28] *IEEE Standard for Information Technology-Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks-Specific*

Requirements Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY)Specifications Amendment1: Enhancements for High-Efficiency WLAN, IEEE, 2021. [Online]. Available: https://dx.doi.org/10.1109/ieeestd.2021.9442429

[29] *3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Requirements for Evolved UTRA (E-UTRA) and Evolved UTRAN (E-UTRAN) (Release 8)*, 3GPP TR 25.913 V8.0.0 (2008-12), 3GPP, 2008. [Online]. Available: https://www.3gpp.org/specifications-technologies/releases/release-15

[30] *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Release 15 Description; Summary of Rel-15 Work Items (Release 15)*, 3GPP TR 21.915 V15.0.0 (2019-09), 3GPP, 2019.

[31] C. A. Gutierrez, O. Caicedo, and D. U. Campos-Delgado, "5G and Beyond: Past, Present and Future of the Mobile Communications," *IEEE Latin America Transactions,* vol. 19, no. 10, pp. 1702–1736, 2021, doi: 10.1109/tla.2021.9477273.

[32] M. T. Moayyed, F. Restuccia, and S. Basagni, "Comparative Performance Evaluation of mmWave 5G NR and LTE in a Campus Scenario," presented at the 2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall), 2020.

[33] R. A. K. Fezeu *et al.*, "An In-Depth Measurement Analysis of 5G mmWave PHY Latency and Its Impact on End-to-End Delay," presented at the Passive and Active Measurement: 24th International Conference, PAM 2023, Virtual Event, March 21–23, 2023, Proceedings, 2023. [Online]. Available: https://doi.org/10.1007/978-3-031-28486-1_13.

[34] *IEEE Standard for the Performance Assessment of Industrial Wireless Systems*, I. SA, 2025. [Online]. Available: https://dx.doi.org/10.1109/ieeestd.2025.11071366

[35] J.-F. Dollinger, K. Bouhouch, and I. Bouzarkouna, "Benchmarking OpenStack for edge computing applications," in *2023 IEEE/ACIS 8th International Conference on Big Data, Cloud Computing, and Data Science (BCD)*, 2023 2023: IEEE, doi: 10.1109/bcd57833.2023.10466323. [Online]. Available: https://dx.doi.org/10.1109/bcd57833.2023.10466323

[36] S. George *et al.*, "OpenRTiST: End-to-End Benchmarking for Edge Computing," *IEEE Pervasive Computing,* vol. 19, no. 4, pp. 10–18, 2020, doi: 10.1109/mprv.2020.3028781.

[37] L. Peroni and S. Gorinsky, "An End-to-End Pipeline Perspective on Video Streaming in Best-Effort Networks: A Survey and Tutorial," *ACM Computing Surveys,* vol. 57, no. 12, pp. 1–47, 2025, doi: 10.1145/3742472.

[38] J. Žádník, M. Mäkitalo, J. Vanne, and P. Jääskeläinen, "Image and Video Coding Techniques for Ultra-low Latency," *ACM Computing Surveys,* vol. 54, no. 11s, pp. 1–35, 2022, doi: 10.1145/3512342.

[39] H. Schulzrinne, S. L. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550, 2003. [Online]. Available: https://www.rfc-editor.org/info/rfc3550

VITA

Benjamin A. Clark was born and raised in Tennessee of the United States. He received his Bachelor of Science in Computer Engineering from the University of Tennessee, Knoxville, in 2023. During his undergraduate studies, he developed a strong interest in embedded systems, wireless communication, and real-time computing, which motivated his pursuit of graduate research in edge and networked systems.

In 2023, he joined the Department of Electrical Engineering and Computer Science at the University of Tennessee as a master's student, where he conducted research in wireless edge computing, network benchmarking, and real-time system performance analysis. His work focused on developing a unified benchmarking framework to evaluate how processing platforms and wireless technologies such as Wi-Fi 6, 4G LTE, and 5G influence latency-sensitive edge applications.

Chapter Seven During his time as a graduate student, Benjamin served as a research assistant in the UTK AT&T 5G Laboratory, contributing to several projects involving wireless networking and critical edge infrastructure. His research was supported by the National Science Foundation Industry–University Cooperative Research Center (NSF IUCRC) under the Center for High-frequency and Circuits for Communication Systems (CHECCS) program.

Benjamin wishes to express his sincere gratitude to his advisors, Dr. Aly Fathy and Dr. Mongi Abidi, for their invaluable mentorship, insight, and encouragement throughout his studies. Their guidance was instrumental in shaping both the technical rigor and practical relevance of this work.

Upon completion of his Master of Science degree in Electrical Engineering at the University of Tennessee, Benjamin plans to continue contributing to the fields of wireless communication, embedded systems, and networked edge intelligence—pursuing innovations that strengthen the reliability, performance, and security of future connected systems.