University of St. Gallen

School of Computer Science

to obtain the title of

Bachelor of Science in Computer Science

Bachelor Thesis on

Feeling the Room

A Practical Study on WiFi-based Occupancy Detection

Submitted by:

Simon Sigg

23-613-334

Approved on the application by:

Prof. Dr. Bruno Rodrigues

Date of Submission:

May 19, 2026

Acknowledgements

I would like to express my sincere gratitude to my thesis advisor, Prof. Dr. Bruno Rodrigues, for his invaluable guidance and constant support throughout the writing process. His feedback and encouragement were instrumental in shaping this work.

I am also deeply grateful to Karim Khamaisi for his continuous support and helpful guidance, which greatly contributed to the completion of this thesis.

Finally, I would like to thank my family and my friends for always being there for me and supporting me.

St. Gallen, May 19, 2026 Simon Sigg

Abstract

WiFi sensing offers a privacy-respecting alternative to cameras and wearable devices for indoor occupancy and localization, since every commodity WiFi chip already estimates the per-subcarrier Channel State Information (CSI) that summarizes how its packets have propagated through the environment. However, a decade of laboratory results has left it unclear how such systems behave on commodity hardware in realistic rooms when no labeled per-site training data is available. This thesis develops and evaluates one such system: ten ESP32-S3 sensing nodes coordinated under a TDMA schedule by a centralized backend, which reduces each CSI vector to a per-link variance feature, compares it against an online Welford baseline learned from the empty room, and combines the active links by intersecting their line segments into a two-dimensional position estimate together with an explicit uncertainty radius, or an explicit null output when the available evidence is insufficient. No labeled occupant data enters the pipeline at any point.

The system is evaluated in 30 annotated trials recorded in a $7 \times 4.4\,\mathrm{m}$ indoor environment, covering five static poses and a moving polyline under frozen and unfrozen baselines. It operates as a usable occupancy detector ($F_1 = 93.9\%$) and as a coarse localizer (median error $0.84\,\mathrm{m}$ static at 85% coverage, $1.35\,\mathrm{m}$ moving at 96% coverage). The evaluation further isolates two structural limitations: the empty-room reference drifts out of validity within the span of a single trial, and per-pose accuracy is set by mesh geometry rather than by filter tuning. Both are addressable through placement-aware deployment and bounded-weight baseline adaptation, rather than reflecting fundamental limits of CSI sensing on commodity hardware.

Table of Contents

List of Abbreviations

ADC	Analog-to-Digital Converter
AGC	Automatic Gain Control
AoA	Angle of Arrival
AP	Access Point
API	Application Programming Interface
AUC	Area Under the (ROC) Curve
CDF	Cumulative Distribution Function
CFO	Carrier Frequency Offset
CFR	Channel Frequency Response
CNN	Convolutional Neural Network
CRC	Cyclic Redundancy Check
CSI	Channel State Information
DWT	Discrete Wavelet Transform
ESP-IDF	Espressif IoT Development Framework
EWMA	Exponentially Weighted Moving Average
FFZ	First Fresnel Zone
FPR	False Positive Rate
HT20	High Throughput 20 MHz (802.11n channel width)
HVAC	Heating, Ventilation, and Air Conditioning
IoT	Internet of Things
IP	Internet Protocol

IQ In-phase / Quadrature

IQR Inter-Quartile Range

LE Little Endian

LTS Long Training Symbol

MAC Media Access Control

MAD Median Absolute Deviation

MIMO Multiple-Input Multiple-Output

NIC Network Interface Controller

OFDM Orthogonal Frequency Division Multiplexing

PCA Principal Component Analysis

RF Radio Frequency

RSSI Received Signal Strength Indicator

RX Receiver

SDK Software Development Kit

SFO Sampling Frequency Offset

SISO Single-Input Single-Output

SRAM Static Random-Access Memory

STO Symbol Timing Offset

TCP Transmission Control Protocol

TDMA Time-Division Multiple Access

ToF Time of Flight

TPR True Positive Rate

TX Transmitter

UDP User Datagram Protocol

WiFi Wireless Fidelity (IEEE 802.11 wireless networking)

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

Smart buildings and ambient assisted living increasingly depend on reliable occupancy information to optimize energy consumption, enhance security, and support elderly care. Conventional methods rely on cameras or wearable devices. However, these approaches either introduce substantial privacy concerns or depend on active user participation [36, 41].

WiFi-based sensing offers a different trade-off. Every commodity WiFi chip already estimates the wireless channel on every packet, producing a channel state information (CSI) vector per-subcarrier that summarizes how the signal has propagated through the environment [14]. A person moving through or standing in that environment perturbs multipath propagation and leaves a measurable trace in CSI [20, 33]. From these perturbations, the literature has demonstrated occupancy detection, people counting, activity recognition, coarse localization, and even vital-sign monitoring, all without cameras or wearable devices [17, 36, 39, 41]. The data stream carries no images, no audio, and no biometrics, only an aggregated low-rate measure of how the channel has changed.

Despite a decade of promising laboratory results, real-world deployment of WiFi sensing remains challenging [26]. Systems trained in one environment routinely fail in another due to hardware heterogeneity, environmental variability, and the difficulty of separating genuine human activity from multipath artefacts [29, 32, 40]. Beyond cross-environment generalization, the practical limits of such systems – detection range, robustness to interference, and stability over time – remain inadequately characterized [2, 26, 37]. This thesis therefore focuses on systematically evaluating a WiFi sensing system built for practical deployment on commodity hardware and operated without per-site labeled training data.

1.1 Overview

This thesis develops and evaluates a WiFi CSI-based occupancy detection and localization system built on a mesh of Espressif ESP32-S3 sensing nodes coordinated by a single processing backend. The nodes rotate through the transmitter and receiver roles under a TDMA schedule, so that every directed link in the mesh contributes CSI measurements once per cycle. The backend reduces each raw CSI vector to a scalar variance feature, smooths it, and compares it against a per-link statistical baseline learned online from the empty room. Links that deviate significantly are declared active; the room is declared occupied once a quorum of links agrees. Active links are then translated into a two-dimensional position estimate by intersecting the line segments they span and taking a weighted centroid of the surviving intersections.

Labeled occupancy data are not required at any point. The system learns its reference from the unoccupied room and reports a position estimate together with an explicit uncertainty radius and an explicit null output when the available evidence is insufficient. The complete pipeline, from the firmware of the sensing node to the live monitoring interface, is implemented from beginning to end and is used to record the annotated trials in a real indoor environment for evaluation in Chapter 5.

1.2 Thesis Goals

The thesis is organized around the following research question.

> *What are the practical capabilities and limitations of WiFi CSI-based occupancy detection and localization on commodity, low-cost hardware in realistic indoor environments, when the system operates without per-site labeled training data?*

Answering this question requires both a system to study and a setup in which it can be studied. The thesis therefore pursues five concrete objectives.

1. **Adapt the inherited sensing pipeline,** Reshape the end-to-end pipeline so that it learns its reference from the unoccupied room and translates per-link channel deviations into a room-level occupancy decision together with a two-dimensional position estimate and an associated uncertainty.

2. **Deploy the sensing system on ESP32-S3 hardware.** Establish a reproducible deployment methodology that covers firmware adaptation, mesh network configura-

tion, and calibration procedures, so that the system can be consistently brought up on commodity nodes.

3. **Empirically evaluate the system in a realistic indoor environment,** Collect annotated trials covering both static and moving occupants under reproducible conditions, and characterize the system's behavior along three axes: detection accuracy, localization error, and coverage, that is, the fraction of occupied time for which any position estimate is produced.

4. **Identify and characterize the system's failure modes,** and analyze the root causes of false positives, missed detections, and degraded localization, so that the evaluation produces concrete starting points for future improvements rather than aggregate metrics alone.

5. **Assess the gap between research prototype and practical deployment,** and distill the empirical findings into evidence-based recommendations for moving WiFi sensing from laboratory demonstrations to real-world indoor applications.

These objectives are interdependent. Design choices are motivated by gaps in related work. The implementation determines what the evaluation can observe, and the evaluation determines which design choices were good and which were not. The contribution of the thesis is the combination of design, implementation, and evaluation, rather than any single one in isolation.

1.3 Organization

The remainder of the thesis is organized as follows. Chapter 2 introduces the necessary background on WiFi CSI, surveys related work in WiFi sensing, and identifies the gaps that this thesis addresses. Chapter 3 develops the design of the system, stating its objectives and walking through the signal processing pipeline, the centralized architecture, the calibration subsystem, and the geometry-based localizer. Chapter 4 translates these design decisions into concrete implementation choices, covering the hardware and firmware of the sensing node, the binary protocol, and the backend processing pipeline. Chapter 5 reports the experimental results, evaluating the accuracy, localization error, and coverage of the trials recorded in a real indoor environment. Chapter 6 concludes by relating the results back to the research question and discussing the limitations of the present system and the directions for future work.

Chapter 2

Fundamentals

Chapter 2 establishes the technical foundations on which the rest of the thesis is based and situates the work within the existing literature. Section 2.1 first introduces the wireless communication concepts that make WiFi sensing possible, walks through the signal processing steps required to turn raw Channel State Information (CSI) into a usable sensing feature, and surveys the main paradigms of WiFi-based indoor localization. Based on this background, Section 2.2 then reviews the most relevant prior CSI sensing systems, compares them along the dimensions of hardware, signal processing, and localization method, and identifies the research gaps that motivate the design choices presented in Chapter 3.

2.1 Background

2.1.1 Wireless Communication Fundamentals

WiFi sensing repurposes the physical-layer signals of standard WiFi infrastructure to passively perceive the environment. Several interlocking communication fundamentals make this possible [20].

2.1.1.1 Multipath Propagation

WiFi signals do not travel in a single straight line, but scatter, reflect, and diffract off walls, furniture, and human bodies, arriving at the receiver via multiple paths simultaneously. The presence and movements of humans alter the signal propagation paths, leading to measurable changes in the received signal. These variations can be used for WiFi sensing to interpret human movements [20].

2.1.1.2 Fresnel Zone

In WiFi, Fresnel zones refer to a set of concentric ellipsoids with foci on a pair of transceivers P_1 and P_2. For a given radio wavelength λ, the n-th Fresnel zone is con-

structed by ensuring that the total path length via any point Q_n on its boundary satisfies [30]:

$$|P_1Q_n| + |Q_nP_2| - |P_1P_2| = \frac{n\lambda}{2} \tag{2.1}$$

The innermost ellipsoid is defined as the 1st Fresnel zone, with successive zones forming elliptical rings around it. More than 70% of the transmitted energy is carried within the first Fresnel zone, making it the dominant contributor to the received signal [30].

2.1.1.3 Orthogonal Frequency Division Multiplexing (OFDM)

Modern WiFi standards rely on Orthogonal Frequency Division Multiplexing (OFDM) to transmit data across indoor wireless channels. OFDM divides a wide-band channel into many narrow-band subcarriers that are mathematically orthogonal, enabling efficient parallel data transmission while mitigating frequency-selective fading. This allows for per-subcarrier channel measurements rather than a single aggregate signal, which is critical for sensing purposes [4, 20].

2.1.1.4 Channel State Information (CSI)

Channel State Information (CSI) is the primary sensing signal extracted from WiFi. It describes the Channel Frequency Response (CFR) of the wireless medium. For each subcarrier frequency, the channel response is modeled as a superposition of contributions from all propagation paths [20]:

$$H(f;t) = \sum_{n}^{N} a_n(t) e^{-j2\pi f \tau_n(t)} \tag{2.2}$$

were $a_i(t)$ denotes the amplitude attenuation factor and $\tau_i(t)$ the propagation delay of the i-th path at time t, while f is the carrier frequency. Both the CSI amplitude $|H|$ and the phase $\angle H$ are affected by any movement of the transmitter, receiver, or surrounding objects, making the CSI a sensitive indicator of environmental changes [20].

CSI is estimated by the receiver using known reference signals embedded in the WiFi packet preamble, called Long Training Symbols (LTS). For each subcarrier, the transmission can be described by the linear model: $y = Hx + n$ where y is the received signal, x is the known transmitted symbol, H is the channel matrix to be estimated, and n represents the noise. By comparing x and y, the receiver computes an estimate of H for each subcarrier in each pair of antennas [20].

For a WiFi system with MIMO-OFDM with M transmit antennas, N receive antennas, and K subcarriers, the resulting CSI snapshot at a given point in time is a complex-valued three-dimensional matrix $H \in \mathbb{C}^{(NMK)}$.
Collecting these snapshots over time produces a four-dimensional CSI tensor that encodes channel variation across spatial, frequency, and temporal domains simultaneously [20].

2.1.2 Signal Processing

Raw CSI measurements contain substantial noise that must be addressed before meaningful sensing is possible [20]. This section reviews the signal processing pipeline that transforms noisy CSI streams into features suitable for occupancy detection.

2.1.2.1 Automatic Gain Control

A fundamental preprocessing step when working with raw CSI data is the compensation of Automatic Gain Control (AGC) effects. The AGC circuit in a wireless receiver dynamically scales the amplitude of the incoming signal before analog-to-digital conversion, ensuring that the signal remains within the operational dynamic range of the ADC's and preventing both clipping and quantization noise [18, 34].

Because this gain adjustment occurs before channel estimation, it introduces a packet-varying multiplicative offset into the measured CSI amplitude that is independent of the true underlying channel conditions [24]. Consequently, the raw CSI amplitude does not faithfully represent the actual channel attenuation, which can severely degrade sensing accuracy if left uncorrected [34]. To mitigate this, the AGC gain value reported by the NIC for each received packet is used to normalize the raw CSI, effectively dividing the hardware-induced offset and recovering an estimate of the true channel frequency response [5].

2.1.2.2 Phase Sanitization

Another critical preprocessing step is phase sanitization, which addresses the systematic and random phase errors introduced by hardware imperfections in commodity Wi-Fi NICs. Phase noise in raw CSI generally originates from two sources: random environmental factors and hardware device imperfections, with imperfect synchronization between transmitter and receiver giving rise to several different error types, including Sampling Frequency Offset (SFO), Symbol Timing Offset (STO), and Carrier Frequency Offset (CFO) [33].

The SFO arises from a mismatch between the oscillators at the transmitter and receiver, generating a time shift of the received signal relative to the transmitted one. STO occurs because the receiver's packet detection process introduces a random time shift due to hardware imperfection. CFO occurs because the receiver's local oscillator cannot be perfectly synchronized with the transmitter's carrier frequency [7].

Crucially, SFO and STO both manifest as linear phase distortions across subcarriers, meaning the raw phase-frequency relationship of the CSI is corrupted by an unwanted linear slope and offset that are unrelated to the physical channel. A linear transformation of the phase is commonly applied to correct these linear offsets, and in some cases additional filtering in time or frequency is performed to further smooth the data [7].

2.1.2.3 Outlier Removal

Following hardware-induced amplitude and phase corrections, a further sensible pre-processing step is the removal of outliers from the raw CSI time series. Raw CSI data collected from commodity network interface cards typically contain considerable noise and outliers caused by ambient factors and hardware devices, and therefore the signal pre-processing step is necessary to eliminate them before the data can be used for sensing [33]. These outliers may manifest due to noise and interference in the wireless channel [13] and, if left uncorrected, these high-energy impulses and bursts mask the subtle, slowly-varying amplitude changes induced by human motion that sensing algorithms rely upon [19].

One way to remove such outliers in real time is the Hampel filter [13]. It operates by sliding a window of configurable width over the time series and computing a local median and a robust estimate of dispersion using the Median Absolute Deviation (MAD) for each sample. Any sample that deviates from the local median by more than a defined multiple of the MAD is identified as an outlier and replaced with the median value [27]. This approach is preferred over simpler statistical methods because the use of the median and MAD renders it inherently robust to the very outliers it seeks to detect, a property that mean- and standard-deviation-based methods do not share [27].

2.1.3 Wi-Fi-based localization

Wi-Fi-based localization uses measurements of Wi-Fi signal characteristics to estimate the position of a person or device [38]. This section shows some possible approaches.

2.1.3.1 Fingerprinting

Fingerprinting treats localization as a pattern-matching problem and proceeds in two phases. During an offline phase, signal measurements are collected at a set of reference points with known coordinates and stored in a radio map, together with their location labels. During the online phase, a newly observed measurement is compared against this database, and the location is estimated from the best-matching entries [12].

2.1.3.2 Geometry-based

Geometry-based approaches estimate position directly from physical signal parameters using the geometric relationships between transmitters and receivers, without requiring a pre-recorded database. Distances are commonly inferred from the time of flight (ToF) or from path loss models, angles from antenna arrays measuring the Angle of Arrival (AoA), and the position is then obtained through trilateration or triangulation [23].

2.2 Related Work

This section reviews existing research and methodologies relevant to device-free localization and CSI-based WiFi sensing. It examines prior approaches to identify the most applicable strategies for the proposed system, while highlighting the gaps and considerations that motivate the design decisions made in this thesis. Building on the technical foundations introduced in the preceding chapter, WiFi sensing can be applied to detect and localize human presence without requiring any device attached to the target. Prior work shows that CSI-based WiFi sensing systems still struggle to combine commodity hardware, minimal or no environment-specific training, and robust performance across diverse real-world deployments [3, 21].

2.2.1 Review of Existing Systems

Table 2.1 presents a comparison of the ten most relevant studies for this thesis. The table focuses on the most critical aspects of the sensing pipeline: the hardware platform, the signal-processing approach applied to the raw CSI, the localization or detection method, the sensing task, and the primary objective of each study. This overview aims to clarify which aspects of prior work have been successful, where challenges persist, and how each contribution relates to the goals of this system.

Table 2.1: Overview of CSI-based Wi-Fi sensing studies.

Study	Hardware	CSI Processing	Method	Sensing Task	Objective
[35]	Intel 5300 NIC	Subcarrier averaging, CSI_{eff} metric	Trilateration	Device-based localization	Fine-grained localization using CSI in OFDM
[10]	Intel 5300 NIC	PCA, Hampel filter, wavelet transform	Adaptive threshold, link crossing geometry	Localization / Tracking	Wi-Fi grid based location tracking
[15]	ESP32	DWT, PCA	Dense neural network	Activity Recognition / Presence Detection	Lightweight IoT-based WiFi sensing on commodity hardware
[3]	Asus RT-AX86U (Wi-Fi 6)	Doppler features (SHARP)	CNN-based classifier	Activity Recognition	Systematic investigation of Wi-Fi 6 sensing capabilities and limitations
[21]	ESP32-S2 WROOM	CSI amplitude, outlier removal, wavelet transform	CNN-based classifier	Activity Recognition	Temporal stability of CSI data for activity recognition
[6]	Intel 5300 NIC	Wavelet domain denoising, phase sanitization	Fingerprint matching	Localization	Indoor localization using combined CSI amplitude and phase
[31]	Intel 5300 NIC	Subcarrier selection (FFZ model)	Model-based CSI equations, Genetic Algorithm	Localization	Device-free localization with some unknown transceiver positions
[17]	Intel 5300 NIC	Joint AoA–ToF super-resolution	Likelihood-weighted multi-AP localization	Localization	Decimeter indoor localization using commodity WiFi APs
[16]	ESP32-C3	CSI amplitude, per-link Z-score	AUC-based link selection, naturalistic study	Presence Detection	Demonstrates that denser multi-link meshes can degrade accuracy due to uninformative links diluting the signal from well-placed ones

The WIDE system [10] follows a detection-and-intersection approach instead of model fitting. Each WiFi link is treated as an infrared-style beam, and a threshold derived from the static-environment CSI eigenvalue distribution is applied per link to flag obstruction events. The target position is inferred from the geometry of simultaneously obstructed links, achieving around 0.95 m average tracking error in a 3×3 transceiver grid with no offline training phase [10]. Two aspects of WIDE are close to the approach taken in this thesis: the use of a statistical deviation metric rather than a learned classifier, and the use of link intersection geometry for position estimation. A notable limitation is that WIDE requires knowledge of the target's starting location and assumes continuous movement between grid cells, which the present system avoids.

FILA models the relationship between a target's position and the aggregate CSI amplitude across 30 subcarriers using a path-loss-inspired CSI_{eff} metric and achieves median errors of about 0.5 m in laboratory settings [35].

LiFS demonstrate device-free localization without a full site survey. LiFS formulates localization as a set of power-fading equations, one per WiFi link, and solves directly for the target coordinates, reaching 0.5 m median accuracy with 11 laptops in line-of-sight conditions [31].

A recent naturalistic study by Rodrigues and Khamaisi [16] directly challenges the assumption that denser link meshes yield better detection accuracy. In a 12-day residential deployment using a 9-node ESP32-C3 mesh with 72 sensing links, a single well-placed link outperformed the full mesh (AUC 0.541 vs. 0.489). The authors attribute this to a dilution effect: links whose Fresnel zones do not intersect the activity region contribute noise that overwhelms the signal from informative links. Critically, strategic link placement mattered more than classifier choice, and random link selection matched optimized selection.

SpotFi [17] and Dang et al. [6] provide higher accuracy at the cost of higher complexity. SpotFi applies a super-resolution method to jointly estimate AoA and ToF for each propagation path. Then it identifies the direct path at each access point and localizes by determining which one is the most likely, achieving 40 cm median error in an indoor office deployment [17]. Dang et al. show that combining calibrated CSI phase information with amplitude fingerprints outperforms amplitude-only approaches, with about 87% of test positions within 1 m [6]. The phase sanitization used in that work, which removes the linear phase offset from hardware clocks, directly relates to the phase processing concepts discussed in the background chapter. However, both methods require

either offline fingerprinting or careful per-deployment configuration, which creates a deployment barrier that this thesis aims to avoid.

Two works are particularly relevant for the hardware and sensing model of this thesis.

Hernandez & Bulut [15]: This work characterizes the ESP32 microcontroller as a standalone CSI sensing node. A low-cost battery-powered ESP32 running the ESP-IDF framework can collect 64-subcarrier CSI at up to 650 Hz, without any host computer. It demonstrates that the ESP32 is a suitable platform for the multi-node architecture of this thesis.

Matey-Sanz et al. [21]: This study quantifies how trained CSI models degrade over time. A CNN trained for activity recognition on ESP32-collected data loses 52% accuracy after 10 minutes and 73.6% after 90 minutes. The authors attribute this to slow environmental drift, such as temperature changes and shifting interference, which continuously shift the CSI distribution even when the room itself does not change.

2.2.2 Research Gaps & Considerations

Despite meaningful progress in the reviewed literature, several gaps remain that are directly relevant to the goals of this thesis.

- **Calibration and generalization** Multiple systems require an offline setup phase that must be repeated for every new environment. [3] confirms that trained models struggle to generalize across rooms, and [21] shows that even a short time after training, accuracy degrades significantly. A deployment-ready system that requires no prior data collection remains largely absent from the literature.

- **Adaptive baselines** Systems that rely on a fixed CSI reference model are sensitive to environmental drift [21]. A continuously updated statistical baseline, one that absorbs gradual changes while remaining sensitive to human presence, could improve robustness in real-world conditions.

- **Dense commodity-hardware meshes** Works that benefit from many nodes, such as [31] often depend on bulky hardware. ESP32 enables small, low-cost, standalone nodes, but dense meshes introduce a dilution risk: Rodrigues and Khamaisi [16] show that non-informative links can degrade aggregate detection performance. Node placement and link selection therefore matter as much as mesh density.

- **Presence localization without training data** Existing surveys indicate that most CSI-based WiFi sensing systems focus on activity recognition and related fine-

grained human sensing tasks, typically relying on supervised learning or deep models[20, 26, 36]. The simpler task of detecting and approximately localizing a person using only link geometry has received comparatively little attention on commodity hardware.

2.2.2.1 Key Insights for Methodology

Building on the reviewed literature and the identified gaps, the methodology of this thesis is guided by the following considerations.

- A rolling Z-score per link can serve as the detection primitive, with the mean and variance of the CSI amplitude of each link updated online using standard incremental estimators such as Welford's algorithm [28]. This continuous update allows the baseline to track slow environmental drift without any explicit recalibration phase, directly addressing the temporal instability observed in prior CSI sensing studies [2, 21].
- Treating obstructed links as geometric constraints and deriving a position estimate from their intersection region is aligned with prior work such as [10], which uses link geometry rather than training data to infer target location.
- Not all links contribute equally. The dilution effect [16] motivates treating node placement as an experimental variable.

Chapter 3

Design

Chapter 3 describes the design of the WiFi-based occupancy detection and localization system developed in this thesis. The system is shaped by the gaps identified in the related-work review. At the same time, the design must produce evidence of a quality sufficient to answer the research question about the practical capabilities and limitations of such a system.

Figure 3.1 shows the per-link signal processing pipeline, from raw CSI capture at the sensing nodes to the room-level occupancy decision. The chapter follows it stage by stage and then describes how the resulting active links are translated into an estimated location. Section 3.1 fixes the objectives. Section 3.2 introduces the centralized architecture in which the pipeline runs. Sections 3.3 and 3.4 cover data collection from the node mesh and the scheduling discipline that prevents collisions. Sections 3.5 through 3.8 cover the central processing chain from raw CSI to the estimated location. Sections 3.9 and 3.10 describe the experimental scaffolding that supports the evaluation in later chapters.

The focus throughout is on what the system is and why each component takes the form it does. Implementation details are deferred to Chapter 4.

3.1 Design Objectives

The research question asks what the practical capabilities and limitations of WiFi-based occupancy detection are on commodity hardware in realistic indoor environments. Answering it requires a system whose properties are shaped by the gaps identified in Section 2.2, and a methodological setup that allows these capabilities and limitations to be observed.

3.1.1 System Properties

The first group of objectives concerns properties that the system itself should exhibit.

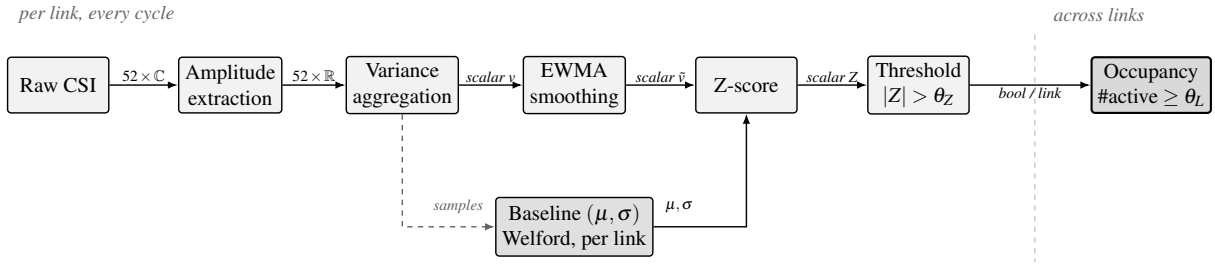

Figure 3.1: Signal processing pipeline. Raw CSI from each link is reduced to a scalar Z-score against a per-link baseline learned online by Welford's algorithm. Stages left of the dashed separator operate independently per link; the final stage fuses link activations into the room-level occupancy decision. The dashed feedback arrow indicates that the baseline learns from the raw variance scalar; the Z-score stage then evaluates the smoothed variant of the same signal. It may be frozen on entering monitoring mode.

1. **Training-free operation;** The system must be able to estimate the location of a person, without a long offline training phase or manual site survey.
2. **Drift resilience;** The reference against which deviations are measured should absorb slow environmental change, such as temperature, humidity, or shifting interference, without losing sensitivity to human presence.
3. **Multi-node operation on commodity hardware;** The architecture must support a mesh of low-cost, standalone sensing nodes rather than a small number of host-tethered NICs.
4. **Reproducible recording;** To be able to evaluate the system, measurements must be stored in a structured, annotated form supporting offline re-analysis.

Objectives 1–3 drive decisions about the sensing layer, the calibration subsystem, and the localization. Objective 4 drives decisions about the data collection subsystem.

3.2 Overall Architecture

A single processing backend acts as the sole coordinator (Figure 3.2); all sensing nodes communicate exclusively with it. Nodes do not exchange messages with each other at the application layer. This centralized approach keeps the coordination logic simple.

The architecture comprises three layers:

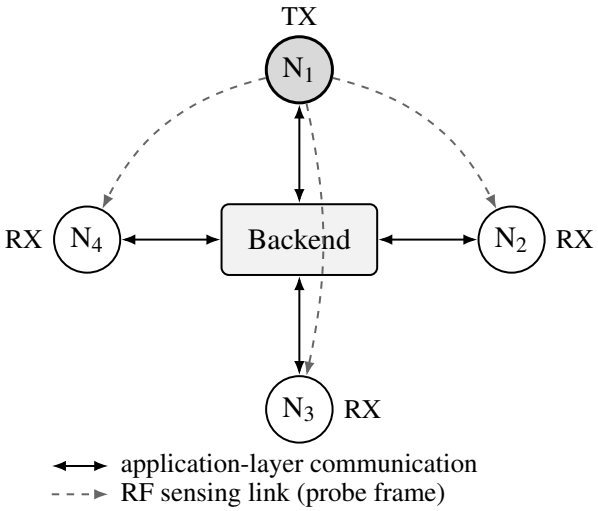

Figure 3.2: Centralized architecture, snapshot of one TDMA slot. Nodes communicate only with the backend (solid arrows). N_1 is transmitting; the others extract CSI from its probe frames (dashed). The TX role rotates across slots.

- **Sensing layer:** a set of distributed wireless nodes, each capable of transmitting probe frames and capturing the resulting CSI on all received frames from designated transmitters.
- **Processing layer:** a backend service responsible for scheduling, receiving, processing, and recording measurements based on which it estimates the position of an occupant.
- **Presentation layer:** a monitoring interface that consumes processed state from the backend and presents it in real time to an operator.

3.2.1 Communication Protocol

All communication between nodes and the backend uses a small set of fixed message types transmitted over the local network. Table 3.1 lists each type, its direction, and its role in the system.

The protocol is intentionally minimal. Nodes never initiate scheduling decisions and never address each other directly; every message either reports node state upward to the backend or carries a scheduling command downward from it. The slot identifier included in both the TRIGGER and the CSI BATCH provides the binding between a scheduling decision and the measurements it produces, which is what allows the backend to attribute

Table 3.1: Message types and their roles.

Message	Direction	Purpose
HELLO	Node → Backend	Sent on boot and periodically thereafter. Announces the node's identity, network address, and hardware information, allowing the backend to register the node and begin scheduling it.
HEARTBEAT	Node → Backend	Sent at a fixed interval. Carries uptime and memory statistics, enabling the backend to detect node failures and monitor health without relying on CSI traffic alone.
TRIGGER	Backend → Node	Broadcast to all nodes at the start of each TDMA slot. Designates which node shall transmit and carries the parameters for that slot. Every node receives the message; only the designated transmitter acts on it by sending probe frames.
CSI BATCH	Node → Backend	Sent by a receiver at the end of each slot. Contains the CSI measurements captured from the designated transmitter's probe frames, together with the slot identifier needed by the backend to resolve which link the samples belong to.

each batch of samples to a specific directed link. The binary encoding and the exact field layout of each message are described in 4.

3.3 Sensing Layer

The sensing layer forms the foundation of the system by defining how individual nodes interact with the wireless environment. It enables the generation of structured channel measurements across the network in a controlled and repeatable manner. By assigning transmitter–receiver roles and collecting CSI, the sensing layer supports fine-grained observation of spatial changes within the monitored space.

3.3.1 Node Responsibilities

From a design perspective, a sensing node is responsible for:

1. Announcing its identity and health to the backend on startup and periodically thereafter.
2. Listening for scheduling commands from the backend.
3. When designated as Transmitter (TX): transmitting a burst of probe frames immediately upon command.

4. When designated as Receiver (RX): capturing all received frames that originate from the current designated transmitter, batching the resulting CSI measurements, and forwarding the batch to the backend.

Roles are not permanent; every node cycles through TX role over successive scheduling rounds. This rotation ensures that every ordered (TX, RX) pair in the network, called a link, is sampled.

3.3.2 Link Coverage

For a network of N nodes, each TDMA cycle produces CSI measurements for all $N(N-1)$ directed links. Each link is an independent sensing channel: the multipath profile between a specific transmitter and a specific receiver is geometrically unique, and different links will be affected by motion in different regions of the room [16].

3.4 Scheduling and Coordination

This section describes how transmissions are scheduled to ensure reliable and interference-free CSI collection. The backend assigns transmission opportunities and structures time into cycles, enabling systematic sampling of all links while adapting to network scale and calibration needs.

3.4.1 Time-Division Multiple Access

Simultaneous transmission by multiple nodes would cause collisions and corrupt CSI at all receivers. The backend therefore enforces a TDMA discipline: in each time slot, exactly one node is designated as transmitter. Slots are arranged in a round-robin sequence over all nodes, with a configurable inter-slot gap to allow receivers to flush and report their captures before the next slot begins.

A single complete round, comprising one TX slot per node, constitutes a cycle. The cycle duration is a function of the number of nodes and the slot duration, and directly determines the per-link sampling rate available to the localization pipeline.

3.4.2 Spatial Zoning

As the network grows, interference between physically proximate nodes becomes a practical concern. The scheduling design supports grouping nodes into zones based on their known positions in the room. Nodes within a zone are scheduled consecutively, ensuring that transmissions from spatially proximate nodes are always separated by at

least one full slot duration. Nodes without known positions are distributed evenly across zones as a fallback.

Zoning affects only the scheduler. The localization pipeline still receives one CSI measurement per link per cycle, whether the TX and RX nodes are in the same zone or in different zones.

3.4.3 Adaptive Slot Allocation

During the initial calibration phase (Section 3.6), different links accumulate baseline samples at different rates, depending on factors such as node proximity, signal quality, and transient obstructions. To accelerate convergence of lagging links, the scheduler supports per-node slot multipliers during the calibration phase. Once the first node reaches its sample target, the scheduler shifts to a weighted cycle: fully calibrated nodes are skipped entirely, while lagging nodes are assigned additional TX slots proportional to how far their worst link is from the target. This mechanism is temporary; once the system transitions to monitoring, uniform round-robin scheduling resumes.

3.5 Signal Processing Pipeline

For each received CSI measurement, the backend executes the following processing steps in sequence.

3.5.1 Amplitude Extraction

The raw CSI vector contains complex-valued (I, Q) channel coefficients for each subcarrier. The amplitude of subcarrier k is:

$$a_k = \sqrt{I_k^2 + Q_k^2} \tag{3.1}$$

Amplitude is used rather than raw complex coefficients because it is invariant to carrier phase offset, which varies across packets due to timing differences [10].

3.5.2 Variance Aggregation

Rather than tracking each subcarrier independently, the pipeline reduces the 52-dimensional amplitude vector to a single scalar per measurement: the variance across subcarriers,

$$v = \frac{1}{K} \sum_{k=1}^{K} (a_k - \bar{a})^2 \tag{3.2}$$

This scalar captures the dispersion of the channel response across the band. Human presence in the Fresnel zone of a link alters the multipath structure [30] and therefore perturbs the relative amplitudes of different subcarriers, which is reflected in a systematic change in v.

3.5.3 Temporal Smoothing

Individual variance measurements are noisy. To suppress transient fluctuations, the pipeline maintains an exponentially weighted moving average (EWMA) per link [22]:

$$\tilde{v}_t = \alpha \cdot v_t + (1 - \alpha) \cdot \tilde{v}_{t-1} \tag{3.3}$$

The smoothing factor $\alpha \in (0, 1)$ controls the effective temporal window. A smaller α gives a longer memory and greater noise rejection at the cost of detection latency; a larger α gives faster response but more variance [22]. The EWMA provides the primary input to the detection stage.

3.6 Calibration Subsystem

The raw amplitude variance of a link depends on the physical environment: room geometry, wall material, furniture, and the positions of the nodes [35]. These factors are unknown at deployment time and differ from site to site. The calibration subsystem learns a per-link statistical baseline from measurements taken while the room is unoccupied, rendering the detection stage environment-agnostic.

3.6.1 Online Baseline Estimation

The baseline for each link is represented by the mean μ and standard deviation σ of its variance signal in the empty-room condition. These statistics are estimated using an online algorithm that maintains exact running mean and variance from a stream of scalar samples without storing the full history [28].

3.6.2 Calibration Readiness

A link is considered calibrated once it has accumulated a minimum number of baseline samples and its empirical sampling rate exceeds a floor threshold, the latter ensuring that links with unreliable connectivity do not block the transition to monitoring. The system transitions from calibration to monitoring once a configurable quorum of viable links

reach calibrated status. This quorum-based criterion prevents a single problematic link from indefinitely blocking detection.

3.6.3 Baseline Freeze

Once the system enters monitoring mode, baseline updates can optionally be frozen. When frozen, the baseline statistics (μ, σ) remain fixed, and the detection stage computes deviations relative to that fixed reference. When unfrozen, the baseline continues to adapt to new measurements, allowing long-term environmental drift (temperature, humidity, furniture changes) to be absorbed without requiring a full recalibration.

This freeze mechanism is essential for detecting stationary occupants. Without it, a motionless person would gradually be absorbed into the evolving baseline statistics, causing the system to cease flagging their presence. This will be shown in Chapter 5.

3.7 Per-Link Z-Score and Occupancy Detection

Given the EWMA variance $\tilde{v}$ and the baseline statistics (μ, σ) for a link, the system computes a Z-score [1]:

$$Z = \frac{\tilde{v} - \mu}{\sigma} \tag{3.4}$$

The Z-score quantifies how many standard deviations the current signal deviates from the empty-room baseline. A large $|Z|$ indicates that the channel on this link has changed, which is consistent with a person intersecting or reflecting the signal along the corresponding propagation path. A link is declared active when $|Z|$ exceeds a threshold θ_Z.

To improve robustness against noise and transient anomalies, a second threshold θ_L is introduced at the system level. The room is only declared occupied when the number of simultaneously active links equals or exceeds θ_L. This additional criterion reduces the likelihood that isolated fluctuations caused by multipath effects or measurement noise lead to false occupancy detections.

3.8 Position Estimation

The localization module translates per-link detection state into a two-dimensional position estimate within the room. A geometry-based method is used rather than a fingerprinting or machine-learning approach. Both would need prior occupancy data [11], which would contradict the training-free objective established in Section 3.1. The geometry-based approach requires no prior occupant data beyond the empty-room calibration.

3.8.1 Physical model

The physical premise is that a person attenuates or perturbs any link whose propagation path they intersect. When the detector marks a link active, it asserts that the channel on that link has deviated significantly from its empty-room baseline, consistent with a person located somewhere in the region around the line segment connecting the two nodes [31]. With a single active link the position can only be constrained to that region. With two or more active links, the intersections of those regions yield point estimates that can be fused into a single position.

3.8.2 Link deduplication

Because the sensing mesh measures every directed pair (transmitter, receiver) independently, both directions of the same physical path may be active simultaneously. Before computing intersections, the module collapses each bidirectional pair into a single undirected link, retaining the direction with the higher Z-score. This prevents the same propagation path from imposing the same spatial constraint twice and artificially inflating the weight of one region.

3.8.3 Intersection-based position inference

Each active link is represented as the line segment between its two node positions. Candidate position estimates are generated by computing the pairwise intersection points of all such segments. Two filters are applied before an intersection is accepted. A *geometric quality filter* rejects pairs whose interior angle falls below a minimum threshold: Nearly parallel links intersect at shallow angles, so small measurement errors can cause large shifts in the computed intersection point. A *spatial validity filter* rejects intersections that fall outside the room boundary, with a small margin to tolerate minor extrapolation.

3.8.4 Weighted centroid

Not all valid intersections carry equal information. Two independent criteria determine how much each intersection contributes to the final position estimate.

Detection confidence. An intersection is only as reliable as its less-certain constituent link. If one link is only marginally above the activation threshold, the crossing it forms carries weak evidence regardless of how strongly the second link is activated. The confidence factor is therefore the *minimum* Z-score of the pair, a conservative lower bound that prevents a barely-active link from borrowing credibility from its partner.

Geometric quality. Two nearly-parallel links intersect at a shallow angle, and their crossing shifts dramatically under small signal perturbations. Perpendicular links constrain the crossing tightly in both spatial dimensions. The sine of the inter-link angle encodes this property smoothly: it approaches zero as links become parallel and reaches its maximum at ninety degrees.

The weight of each intersection is the product of these two factors,

$$w_{ij} = \min(|Z_i|, |Z_j|) \cdot \sin\theta_{ij}, \tag{3.5}$$

where Z_i, Z_j are the Z-scores of the two constituent links and θ_{ij} is the angle between them. The position estimate is the weighted centroid of all valid intersection points. The resulting point is clamped to the room boundary to prevent the estimate drifting outside the known physical space due to measurement noise or minor extrapolation.

3.8.5 Uncertainty estimate

Alongside the position, the localizer reports a confidence radius, computed as the weighted standard deviation of the intersection points from the centroid. A small radius indicates that active links produce tightly clustered estimates. A large radius indicates that the intersection cloud is spatially diffuse, either because the active links are geometrically weak or because the detections are inconsistent.

3.8.6 Coverage and null output

The localizer returns no estimate when fewer than two active links are available, or when no intersection pair survives the geometric and spatial filters. This explicit null output is a deliberate design choice: reporting no position is preferable to reporting a spurious one. It introduces a second performance dimension that must be evaluated separately from positional accuracy: *coverage*, defined as the fraction of occupied-room time for which any estimate is produced. A system that achieves low error but high null-output rate is not operationally useful. Conversely, a system that always produces an estimate but with large error may be misleading. The evaluation in Chapter 5 therefore treats coverage and accuracy as distinct, co-equal metrics.

3.9 Data Collection and Experiment Design

This section describes how data is collected and structured to support reproducible experiments and offline analysis. It defines how measurements, annotations, and metadata

are organized into self-contained trials, along with the tools used to ensure consistent and accurately labeled recordings.

3.9.1 Trial-Centric Recording

The system supports structured data collection for offline re-analysis and for evaluating the localization pipeline. Each recording session is assigned a unique sequential identifier and stored in an isolated directory that bundles all measurements produced during that session: raw CSI frames, calibrated activity features, a time-stamped stream of localization estimates, and an annotation file. A trial metadata document records the experimental conditions associated with the session. These include the activity type, subject identifier, deployment environment, and spatial topology of the sensor nodes. As a result, every trial is fully self-describing and can be analysed without external context.

3.9.2 Annotation Model

Ground-truth labels are associated with time intervals rather than individual packets. Each annotation records a start timestamp, an end timestamp, and an activity label drawn from a two-level taxonomy: a coarse category (e.g. *stationary*, *locomotion*) and a fine-grained activity label (e.g. *sitting*, *walking*). For trials in which the subject's position is known, each annotation additionally embeds the ground-truth spatial coordinates in its metadata, so that localization accuracy can be computed directly from the annotation record without consulting any external reference. Annotations are stored inside the trial directory and share the same clock reference as the CSI measurements, enabling downstream analysis to precisely select the CSI samples that fall within any labeled interval.

3.9.3 Experiment Orchestration

To support repeatable multi-phase data collection protocols, the system provides an experiment runner that drives the operator through a configurable sequence of phases. Each experiment begins with a shared empty-room baseline period, during which the room is unoccupied. The runner then iterates over a list of one or more activity steps. For each step, a positioning interval gives the subject time to move into place before the active measurement window begins. Phase transitions are signalled to the operator via audio cues so that no manual timing is required. Annotations for the baseline period and for each active measurement window are created automatically with precise timestamps, removing the need for post-hoc manual alignment of labels and ensuring that every recorded trial has a consistent internal structure regardless of who conducts it.

3.10 Monitoring and Control Interface

The presentation layer exposes the system's state in real time to an operator . Its responsibilities are logically distinct from the processing layer: it consumes a live broadcast of processed state (node health, per-link statistics, occupancy decision, localization estimate, and calibration progress) and renders it without contributing to any processing or storage logic.

The interface supports the following operator tasks:

- **Spatial configuration.** Nodes are positioned on a scaled floor plan together with room features such as walls, furniture, ground-truth poses, and walking paths . Node positions are consumed by the scheduler for zone computation and are recorded in trial metadata for geometric analysis. The map also overlays the live localization estimate so that the operator can visually verify tracking accuracy during a session.
- **Experiment control.** The operator can start and stop recording trials and assign activity labels . When using the structured experiment runner, the operator initiates the experiment. Subsequent phase transitions are driven automatically by the configured timers.
- **Live diagnostics.** Per-link sampling rates, calibration progress, Z-scores, and the occupancy decision are visible in real time, allowing the operator to verify system health before and during data collection.
- **Baseline management.** The operator can trigger a full recalibration or freeze/unfreeze the adaptive baseline.
- **Experiment metadata setup.** Subjects, deployment environments, and sensor topologies are registered through a dedicated setup panel, providing the named identifiers that are attached to every recorded trial.

Chapter 4

Implementation

Chapter 4 translates the design decisions of Chapter 3 into concrete implementation choices. Section 4.1 describes the sensing node hardware and the backend platform. Section 4.2 covers the binary communication protocol, fulfilling the encoding details deferred from Section 3.2.1. Sections 4.3 to 4.5 cover the calibration pipeline, link activation, and position estimation respectively. The backend is written in Python and structured around an `asyncio` event loop. The sensing nodes run custom firmware on ESP32 microcontrollers and communicate with the backend exclusively over UDP.

4.1 Hardware Platform

The system comprises two hardware tiers: a set of distributed sensing nodes and a centralized processing backend.

4.1.1 Sensing Nodes

Each sensing node is an Espressif ESP32-S3 development board. The ESP32-S3 integrates a dual-core Xtensa LX7 processor running at up to 240 MHz with 512 KB of on-chip SRAM, and a single-band 2.4 GHz 802.11n (WiFi 4) radio with one transmit and one receive antenna (SISO) [9]. The ESP32-S3 is chosen for this project because Espressif's official SDK exposes raw per-subcarrier CSI data through a dedicated callback. Accessing this data requires only standard application-level API calls. No custom driver patching, kernel modification, or proprietary firmware replacement is needed. This makes the ESP32-S3 one of the few widely available commodity chips on which CSI extraction is accessible through the vendor's supported, unmodified SDK [8]. The board has no MIMO capability and no 5 GHz support. All sensing is performed on the 2.4 GHz band [9].

The sensing firmware is written to be portable across the ESP32 product family. The backend identifies the connected chip variant from the one-byte chip-ID field in the

HELLO packet, but all nodes deployed in this study are ESP32-S3 units operating under identical firmware and configuration.

4.1.2 WiFi Operating Mode and CSI Format

All nodes operate in HT20 mode (802.11n, 20 MHz channel). In this mode the OFDM channel is divided into $K = 52$ active subcarriers [8].

Each CSI measurement encodes the per-subcarrier channel frequency response as a pair of signed 8-bit integers (I_k, Q_k) for $k = 1, \ldots, 52$. The 52 pairs occupy 104 bytes of the packet payload. Amplitude is computed from each pair as $a_k = \sqrt{I_k^2 + Q_k^2}$ [8].

Listing 4.1: IQ extraction from raw packet bytes: 52 signed int8 pairs reshaped into a (52×2) float array (`protocol.py`).

```
4.1.1 NUM_SUBCARRIERS = 52
4.1.2
4.1.3 def _parse_iq(raw: bytes) -> np.ndarray:
4.1.4     iq_raw = np.frombuffer(raw[:NUM_SUBCARRIERS * 2], dtype=np.int8)
4.1.5     return iq_raw.reshape(NUM_SUBCARRIERS, 2).astype(np.float32)
```

The amplitude vector is computed once on first access and cached on the `CSISample` object to avoid redundant square-root evaluations across the pipeline stages that consume it.

Listing 4.2: Lazy cached amplitude property on `CSISample` (`protocol.py`).

```
4.2.1 @property
4.2.2 def amplitude(self) -> np.ndarray:
4.2.3     if self._amplitude is None:
4.2.4         self._amplitude = np.sqrt(
4.2.5             self.iq_data[:, 0] ** 2 + self.iq_data[:, 1] ** 2
4.2.6         )
4.2.7     return self._amplitude
```

4.1.3 Backend Processing Platform

The backend runs on a central server (commodity PC running Ubuntu 24.04) connected to the same local network as the sensing nodes. The backend software is written in Python 3.12 and uses the `asyncio` event loop for concurrent UDP reception, scheduling, and WebSocket broadcasting. All numerical processing uses NumPy and operates on small fixed-size arrays. The backend is computationally undemanding: per-sample

processing is dominated by a single variance over 52 subcarriers and constant-time online statistics. Throughput is gated by packet arrival rather than CPU.

4.1.4 Network Configuration

Nodes and backend communicate over a shared local-area WiFi network. Nodes send CSI BATCH, HELLO, and HEARTBEAT packets via UDP to the backend on port 5000. The backend broadcasts TRIGGER packets to the limited broadcast address (255.255.255.255) on port 5001, reaching all nodes on the local segment with a single send call. UDP is preferred over TCP to avoid head-of-line blocking: a lost trigger or batch is simply absent from the pipeline rather than delaying all subsequent traffic. A batch of up to 12 samples occupies at most $8 + 12 \times 107 + 4 = 1296$ bytes, which is below the maximum 1500-byte payload of a single Ethernet frame [25], so the data fits into one frame. The typical batch of 10 samples is 1082 bytes.

4.2 Binary Protocol

All communication between nodes and the backend is carried over UDP using a compact binary encoding. Every packet begins with a one-byte magic value `0xC5` and a one-byte message-type field, and is terminated by a four-byte little-endian CRC-32 checksum computed over all preceding bytes. An incoming datagram is discarded immediately if the magic byte does not match or if the CRC does not match, protecting the pipeline against corrupted packets and stray UDP traffic on the same local network.

4.2.1 Message Types

HELLO is sent by a node on boot and periodically to register its MAC, IP, and firmware version with the server. HEARTBEAT is a lightweight keepalive carrying uptime and free-heap diagnostics. TRIGGER flows in the opposite direction, instructing a node to begin a measurement burst, and CSI V5 BATCH carries the resulting channel samples back to the server.

Table 4.1 lists the message-type codes and the structures they identify.

4.2.2 Trigger Packet

The TRIGGER packet is broadcast by the backend at the start of every TDMA slot. Its primary payload is the MAC address of the node designated as transmitter for that slot, together with a one-byte slot identifier. The slot identifier is embedded by every receiving

Table 4.1: Binary message types.

Code	Name	Total size (bytes)
0x01	HELLO	40
0x02	HEARTBEAT	24
0x10	TRIGGER	32
0x07	CSI V5 BATCH	$8+107n+4$ (n samples)

node into the CSI BATCH it subsequently sends, providing the TX-to-batch binding described in Section 4.2.3. Table 4.2 gives the complete field layout.

Table 4.2: TRIGGER packet field layout (32 bytes).

Bytes	Field	Notes
0	`magic`	0xC5
1	`type`	0x10
2–7	`tx_mac`	6-byte Ethernet MAC of designated TX
8–11	`seq`	uint32 LE, monotone counter
12–13	`burst_count`	uint16 LE; frames to transmit per slot
14–15	`slot_ms`	uint16 LE; slot duration in milliseconds
16	`samples_per_meas`	Legacy field, fixed to 5; ignored by current firmware
17	`send_all`	Legacy field, fixed to 1; ignored by current firmware
18	`slot_id`	0–255, wrapping; links batch to TX MAC
19–27	*reserved*	Zero-padded
28–31	`crc32`	CRC-32 (LE) of bytes 0–27

The scheduler assigns each known MAC address a stable `slot_id` for the lifetime of the session. Stability is important: if a node reconnects mid-session, it receives the same identifier it held before, preserving the batch-to-TX mapping without requiring the backend to issue corrective packets.

4.2.3 CSI v5 Batch Packet

The v5 batch format packs multiple CSI samples into a single UDP datagram to reduce per-packet overhead. The packet begins with an eight-byte header, is followed by n fixed-size 107-byte sample records, and concludes with a four-byte CRC-32 covering the entire datagram.

The TX MAC address is absent from both the header and the per-sample record. The backend resolves it from the `slot_id` using the scheduler's stable mapping. The RX MAC is inferred from the UDP source IP, which was registered when the node's HELLO packet was processed. Together, these two resolution steps attribute each batch to

Table 4.3: CSI V5 BATCH header layout (8 bytes).

Bytes	Field	Notes
0–1	`magic / type`	`0xC5`, `0x07`
2	`slot_id`	Identifies the TX node via scheduler mapping
3	`count`	Number of sample records following the header
4–5	`seq_base`	uint16 LE; sequence number of first sample
6–7	`timestamp_lo`	uint16 LE; lower 16 bits of firmware μs clock

Table 4.4: Per-sample record layout within a CSI v5 batch (107 bytes).

Bytes	Field	Notes
0–1	`delta_us`	uint16 LE; μs offset from batch `timestamp_lo`
2	`rssi`	Signed int8; received signal strength in dBm
3–106	`iq_data`	52 pairs of signed int8 (I_k, Q_k), $k = 1 \ldots 52$

a specific directed link (TX, RX) without embedding redundant address information in every sample.

4.3 Calibration Pipeline

The calibration subsystem maintains a per-link statistical baseline of the empty-room amplitude variance so that the detection stage can measure deviations relative to the site-specific baseline without requiring any labeled training data. Three concerns shape its implementation: numerical stability of the running statistics, robustness to marginal or dead links, and efficient convergence in the presence of heterogeneous link quality.

4.3.1 Welford Online Estimator

Each link's baseline is held by a `LinkBaseline` object that accumulates the running mean μ and the running sum of squared deviations M_2 using Welford's algorithm [28]. Given a new sample v_n and the running state $(\mu_{n-1}, M_{2,n-1})$, the update is:

$$\delta = v_n - \mu_{n-1} \tag{4.1}$$

$$\mu_n = \mu_{n-1} + \delta / n \tag{4.2}$$

$$\delta' = v_n - \mu_n \tag{4.3}$$

$$M_{2,n} = M_{2,n-1} + \delta \cdot \delta' \tag{4.4}$$

The sample variance is recovered as $\hat{\sigma}^2 = M_2/(n-1)$ for $n \geq 2$ [28]. It operates in $O(1)$ memory per link, regardless of how many samples have been observed. Listing 4.3 shows the implementation.

Listing 4.3: Welford online estimator: per-link baseline state and update (`detector.py`).

```
@dataclass
class LinkBaseline:
    count: int   = 0
    mean:  float = 0.0
    m2:    float = 0.0   # running sum of squared deviations

    @property
    def variance(self) -> float:
        if self.count < 2:
            return 0.0
        return self.m2 / (self.count - 1)

    @property
    def std(self) -> float:
        return math.sqrt(max(0.0, self.variance))

    def update(self, value: float):
        self.count += 1
        delta = value - self.mean
        self.mean += delta / self.count
        delta2 = value - self.mean
        self.m2 += delta * delta2
```

The baseline update is called on every incoming CSI sample throughout both calibration and monitoring. It is suppressed only when the operator has frozen the baseline during monitoring. As seen previously in Section 3.6.3, freezing is essential for detecting stationary occupants.

4.3.2 Link Viability

Not all links in the mesh deliver samples reliably. A link that is intermittent or dead cannot contribute useful evidence to calibration or detection, and must not be permitted to block either. The backend tracks the arrival timestamps of CSI samples for every link in a sliding window of `RATE_WINDOW_S` = 5 s and computes the current rate by dividing the

window's sample count by its duration. A link is *viable* when its rate meets or exceeds the floor `MIN_LINK_RATE_HZ` = 0.5 Hz.

Listing 4.4: Sliding-window link-rate and viability predicate, abridged from `detector.py`.

```
4.4.1 RATE_WINDOW_S = 5.0
4.4.2
4.4.3 def _link_rate(self, link_key) -> float:
4.4.4     ts = self._link_timestamps.get(link_key)
4.4.5     if not ts:
4.4.6         return 0.0
4.4.7     cutoff = time.time() - RATE_WINDOW_S
4.4.8     while ts and ts[0] < cutoff:
4.4.9         ts.pop(0)
4.4.10    return len(ts) / RATE_WINDOW_S
4.4.11
4.4.12 def _link_is_viable(self, link_key) -> bool:
4.4.13    return self._link_rate(link_key) >= MIN_LINK_RATE_HZ
```

Viability filtering is applied uniformly at three points in the system: the calibration quorum check (Section 4.3.3), the active-link scan (Section 4.4.2), and the adaptive slot allocator (Section 4.3.4). This ensures that dead or marginal links cannot block calibration completion, contribute spurious evidence to detection, or receive extra TX slots that cannot help them converge.

4.3.3 Readiness Quorum

The system transitions from CALIBRATING to MONITORING once a sufficient fraction of viable links have accumulated a minimum number of baseline samples. A global minimum requirement, demanding that every link reach the sample target, would be held hostage indefinitely by a single marginal link. Instead, the implementation applies a quorum: at least `READINESS_QUORUM_FRACTION` = 70 % of viable links must reach `BASELINE_LEARNING_SAMPLES` = 2000 samples before the transition fires. An additional absolute guard of `READINESS_QUORUM_MIN_LINKS` = 10 viable links prevents an early flip when only a small number of nodes have joined the mesh.

Because the readiness check is invoked from the per-sample hot path but performs a linear scan over all tracked links, it is throttled to run at most once per second. This keeps its amortised cost negligible without delaying the state transition perceptibly.

4.3.4 Adaptive Slot Allocation During Calibration

Links accumulate baseline samples at different rates depending on node distance, signal quality, and transient obstructions. Once the first transmitter's worst viable link reaches `BASELINE_LEARNING_SAMPLES`, continuing to schedule all nodes uniformly wastes cycles on already-ready transmitters. The scheduler queries `get_calibration_weights`, which computes a per-TX-MAC slot multiplier: zero for fully calibrated nodes (skip them entirely), and an integer between one and four for lagging nodes, proportional to how far their worst link remains from the target. Algorithm 4.1 describes the weight computation.

Algorithm 4.1 Adaptive calibration slot weights.

Require: Per-link baselines $\mathscr{B}$;sample target N_{cal}

4.1.1 $worst \leftarrow \{\}$
4.1.2 **for** each viable link $(tx, rx) \in \mathscr{B}$ **do**
4.1.3 $\quad worst[tx] \leftarrow \min\big(worst.\text{get}(tx, N_{\text{cal}}), \mathscr{B}[(tx, rx)].\text{count}\big)$
4.1.4 **if** $\max(worst.\text{values}) < N_{\text{cal}}$ **then**
4.1.5 $\quad$ **return** $\{\}$ ▷ no node done yet; keep uniform scheduling
4.1.6 **for** each $tx \in worst$ **do**
4.1.7 $\quad$ **if** $worst[tx] \geq N_{\text{cal}}$ **then**
4.1.8 $\quad\quad w[tx] \leftarrow 0$ ▷ fully calibrated; skip
4.1.9 $\quad$ **else**
4.1.10 $\quad\quad r \leftarrow (N_{\text{cal}} - worst[tx])/N_{\text{cal}}$
4.1.11 $\quad\quad w[tx] \leftarrow \text{clamp}(\text{round}(4r), 1, 4)$
4.1.12 **return** w

The scheduler replaces its uniform round-robin with a weighted cycle once the first TX completes calibration, for the remainder of the calibration phase: a node with weight w receives w consecutive TX slots per cycle. Once the system transitions to MONITORING, the uniform schedule resumes. The worst-link focus prevents a single dead link from pinning its TX node into permanent reinforcement that cannot help it converge, because non-viable links are excluded from the iteration in line 2, and hence from the per-TX minimum in line 3 of Algorithm 4.1.

4.4 Link Activation

Link activation translates the raw per-subcarrier IQ measurements of an individual CSI sample into a binary active/inactive decision by comparing the smoothed amplitude variance against the learned baseline. The pipeline applies three steps in sequence:

EWMA smoothing of the incoming variance [22], Z-score scoring against the baseline, and threshold comparison.

4.4.1 Per-Sample Processing

When a CSI sample arrives, the backend extracts the per-subcarrier amplitude vector from the IQ data. The amplitude of subcarrier k is computed as:

$$a_k = \sqrt{I_k^2 + Q_k^2} \tag{4.5}$$

and the variance of this 52-element vector constitutes the scalar feature:

$$v = \frac{1}{K} \sum_{k=1}^{K} (a_k - \bar{a})^2, \qquad K = 52. \tag{4.6}$$

The scalar v is then folded into an exponentially weighted moving average maintained per link. With smoothing factor $\alpha = 0.2$:

$$\tilde{v}_t = \alpha \cdot v_t + (1 - \alpha) \cdot \tilde{v}_{t-1} \tag{4.7}$$

giving an effective window of approximately five samples, or roughly 100 ms at a representative per-link rate of ∼50 Hz. The effective window scales inversely with the per-link rate, which itself depends on node count via the TDMA cycle. Listing 4.5 shows both the EWMA update and the conditional baseline accumulation.

Listing 4.5: Per-sample EWMA update and conditional baseline accumulation, abridged from `detector.py`.

```
4.5.1 EWMA_ALPHA = 0.2
4.5.2
4.5.3 def on_csi_sample(self, sample, wall_ts=None):
4.5.4     link_key = (sample.tx_mac, sample.rx_mac)
4.5.5
4.5.6     amp_variance = float(np.var(sample.amplitude))
4.5.7     prev = self._ewma_variance.get(link_key, amp_variance)
4.5.8     self._ewma_variance[link_key] = (
4.5.9         EWMA_ALPHA * amp_variance + (1 - EWMA_ALPHA) * prev
4.5.10    )
4.5.11
4.5.12    if link_key not in self._baselines:
4.5.13        self._baselines[link_key] = LinkBaseline()
```

```
4.5.14
4.5.15      baseline = self._baselines[link_key]
4.5.16      if not (self._baseline_frozen and
4.5.17              self._state == DetectorState.MONITORING):
4.5.18          baseline.update(amp_variance)
```

Note that the baseline is updated with the unsmoothed v_t, not the EWMA $\tilde{v}_t$. The two signals serve different roles: the EWMA suppresses measurement noise to improve Z-score stability, while the baseline learns the long-run distribution of the raw variance. Mixing them would cause the baseline mean to underfit the raw signal distribution.

4.4.2 Z-Score Computation and Active-Link Decision

During monitoring, the backend evaluates the Z-score for each link on demand. Given the current EWMA variance $\tilde{v}$ and baseline statistics (μ, σ) for the link:

$$Z = \frac{\tilde{v} - \mu}{\sigma} \tag{4.8}$$

A link is declared *active* when $|Z| > \theta_Z = 2.5$. Three guards gate the Z-score computation before it is attempted: the baseline must have accumulated at least `WELFORD_MIN_SAMPLES` = 10 observations (ensuring that σ is not under-estimated from too few samples); the baseline standard deviation must exceed a numerical floor of 10^{-9} (preventing division by zero on links whose variance is pathologically constant); and the link must currently be viable. These guards are applied in `get_active_links`(`detector.py`).

4.4.3 Occupancy Decision

The room-level occupancy decision aggregates the individual link activations returned by `get_active_links`. Denoting the set of currently active links as $\mathscr{A}$, the room is declared occupied if and only if:

$$|\mathscr{A}| \geq \theta_L \tag{4.9}$$

with $\theta_L = 3$ by default. This two-level design, per-link threshold θ_Z followed by a count threshold θ_L, reduces the rate of false positives. Isolated multipath anomalies or transient interference typically perturb only one or two links simultaneously. Requiring at least three concurrent activations sharply reduces the probability that such events are mistaken for human presence.

4.5 Position Estimation

The localization algorithm, implemented in `localizer.py`, converts the set of active links returned by the detector into a two-dimensional position estimate. It proceeds in four stages: deduplication of bidirectional link pairs, pairwise line intersection, geometric and spatial filtering, and weighted centroid computation.

4.5.1 Directed-to-Undirected Deduplication

The sensing mesh measures every ordered pair (TX, RX) independently, so both the forward link $(i \to j)$ and the reverse link $(j \to i)$ may be simultaneously active for the same physical propagation path. Including both would impose the same spatial constraint twice, artificially concentrating weight on one region of the room. Before forming any intersections, the algorithm collapses each bidirectional pair into a single undirected link, retaining the direction that carries the higher absolute Z-score:

Listing 4.6: Directed-to-undirected link deduplication (`localizer.py`).

```
4.6.1 undir = {}
4.6.2 for link in active_links:
4.6.3     key = frozenset((link["tx"], link["rx"]))
4.6.4     if (key not in undir or
4.6.5             abs(link["z_score"]) > abs(undir[key]["z_score"])):
4.6.6         undir[key] = link
```

After deduplication, only links for which both endpoints have registered positions are retained. Links involving unpositioned nodes cannot contribute geometry and are dropped.

4.5.2 Line Intersection

Each surviving link is represented as the infinite line through its two node positions. The intersection of lines $\ell_1 = (p_1, p_2)$ and $\ell_2 = (p_3, p_4)$ is found analytically. Writing D for the denominator of the linear system:

$$D = (x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4) \tag{4.10}$$

If $|D| < 10^{-12}$ the lines are parallel and the pair yields no intersection. Otherwise the parameter t locating the crossing along ℓ_1 is:

$$t = \frac{(x_1 - x_3)(y_3 - y_4) - (y_1 - y_3)(x_3 - x_4)}{D} \tag{4.11}$$

and the intersection point is $\mathbf{q} = p_1 + t\,(p_2 - p_1)$.

4.5.3 Geometric and Spatial Filters

Two filters gate each candidate intersection before it contributes to the final estimate.

Geometric quality filter. Nearly parallel links intersect at a shallow angle, so small perturbations in Z-score or node position produce large shifts in the crossing point. A pair is accepted only when the acute angle between its direction vectors satisfies $\theta \geq \theta_{\min} = 20^\circ$. The angle is computed using the absolute value of the dot product:

$$\theta = \arccos\left(\frac{|\mathbf{d}_1 \cdot \mathbf{d}_2|}{\|\mathbf{d}_1\|\,\|\mathbf{d}_2\|}\right) \tag{4.12}$$

The absolute value makes the angle invariant to the choice of direction along each link.

Spatial validity filter. Intersections that fall outside the room boundary, extended by a tolerance margin $\Delta = 0.3\,\text{m}$ on each side, are discarded. This margin absorbs intersections from a real near-wall occupant that fall geometrically just outside the room due to position noise and angular uncertainty, without admitting clearly spurious points far outside the monitored space.

4.5.4 Weighted Centroid

Not all accepted intersections carry equal evidential weight. The weight assigned to intersection $\mathbf{q}_{ij}$, formed by links i and j, combines a detection confidence term and a geometric quality term:

$$w_{ij} = \min(|Z_i|, |Z_j|) \cdot \sin\theta_{ij} \tag{4.13}$$

Both factors are motivated in Section 3.8.4: $\min(|Z_i|, |Z_j|)$ is a conservative lower bound that prevents a strong link from compensating for a barely-active partner, and $\sin\theta_{ij}$ smoothly penalises shallow crossings. Intersections with non-positive weight are discarded. In practice this guard never fires: deduplicated links carry $|Z| > \theta_Z > 0$, and the 20 geometric filter ensures $\sin\theta > 0$, so the product $\min(|Z_i|, |Z_j|) \cdot \sin\theta_{ij}$ is strictly positive. The check is retained as defensive coding against numerical edge cases. The position estimate is the weighted centroid of all surviving intersection points, clamped to the room boundary as specified in Section 3.8.4. Listing 4.7 shows the pipeline: angle and bounds filtering, intersection, weighting, and the final centroid.

Listing 4.7: Intersection accumulation, filtering, and weighted centroid (simplified from `localizer.py`).

```
4.7.1 min_angle_rad = math.radians(20.0)
4.7.2 lo_x, hi_x = -0.3, room_width  + 0.3
4.7.3 lo_y, hi_y = -0.3, room_height + 0.3
4.7.4
4.7.5 points = []   # (x, y, weight)
4.7.6 for i, (p1, p2, z1) in enumerate(segments):
4.7.7     d1 = (p2[0]-p1[0], p2[1]-p1[1])
4.7.8     for p3, p4, z2 in segments[i+1:]:
4.7.9         d2 = (p4[0]-p3[0], p4[1]-p3[1])
4.7.10        angle = _angle_between(d1, d2)
4.7.11        if angle < min_angle_rad:
4.7.12            continue                      # geometric filter
4.7.13        pt = _line_intersection(p1, p2, p3, p4)
4.7.14        if pt is None:
4.7.15            continue
4.7.16        if not (lo_x <= pt[0] <= hi_x and lo_y <= pt[1] <= hi_y):
4.7.17            continue                      # spatial filter
4.7.18        weight = min(z1, z2) * math.sin(angle)
4.7.19        if weight > 0:
4.7.20            points.append((pt[0], pt[1], weight))
4.7.21
4.7.22 total_w = sum(w for _, _, w in points)
4.7.23 cx = sum(x*w for x, _, w in points) / total_w
4.7.24 cy = sum(y*w for _, y, w in points) / total_w
4.7.25
4.7.26 cx = max(0.0, min(room_width,  cx))    # clamp to room
4.7.27 cy = max(0.0, min(room_height, cy))
```

4.5.5 Confidence Radius

The localizer reports an uncertainty estimate alongside the position: the weighted standard deviation of the intersection cloud from the centroid,

$$r = \sqrt{\frac{\sum_{ij} w_{ij}\left[(q_{ij,x}-\hat{x})^2+(q_{ij,y}-\hat{y})^2\right]}{\sum_{ij} w_{ij}}} \tag{4.14}$$

A small r indicates that all active links produce tightly clustered intersections; the estimate is geometrically well-constrained. A large r indicates a diffuse intersection cloud, arising either from shallow-angle crossings (close to the 20° threshold) or from inconsistent detections across the mesh. The radius is rendered as an uncertainty circle around the estimated location in the monitoring interface. Two implementation details depart from the equation: when only a single intersection survives, the radius defaults to 1 m — the variance collapses to zero in this case, which would falsely advertise high confidence from a single point, so a fixed nominal value is used instead; and the reported radius is floored at 0.1 m to keep the uncertainty circle visible in the interface.

Chapter 5

Evaluation

Chapter 5 reports the empirical evaluation of the system whose design and implementation were developed in Chapter 3 and Chapter 4. The research question (Section 1.2) asks for the practical capabilities and limitations of a WiFi CSI sensing system on commodity hardware, operated without per-site labeled training data. Answering that question is not a matter of a single accuracy number. It requires that the system be exercised against ground truth in a realistic environment under conditions that separate the cases in which it is expected to work from the cases in which it is not. The evaluation therefore proceeds in three movements: it establishes a measurement methodology and the metrics it will use (Section 5.1); it documents the setup precisely enough that the results could in principle be reproduced (Section 5.2); and it then characterises the system along the three axes named in the third thesis objective, namely detection, localization error, and coverage (Section 5.3–Section 5.6). The chapter closes by analysing the system's failure modes (Section 5.7) and by relating the evidence back to the five objectives that motivated the work (Section 5.8).

5.1 Methodology and Metrics

The evaluation is based on one basic unit called a trial (defined in Section 3.9). Each trial bundles raw CSI, calibrated per-link features, a stream of localizer outputs, and time-stamped annotations carrying ground-truth geometry. A trial therefore contains, at every moment, both the system's estimate and the true value. The evaluation pipeline is a thin layer on top of these trial bundles: it joins the persisted localizer estimates against the embedded ground-truth windows and reports the discrepancy. The pipeline is implemented in `scripts/eval_localizer.py` for per-trial summaries and `scripts/aggregate_eval.py` for the pooled cross-trial aggregates that this chapter reports.

5.1.1 Scenarios

A trial alternates between three labeled scenarios. The **empty** scenario contains no occupant and admits no ground-truth position. The system is expected to remain silent. The **stationary** scenario places a single occupant at one of five fixed poses S1–S5 (see Section 5.2). The ground truth is a single point. The **moving** scenario has the occupant walking a closed polyline at approximately constant speed. The ground truth is the arc-length-parameterised position along that polyline. These three scenarios correspond directly to the three behaviours the system claims: silence when the room is empty, a stable estimate when a person is still, and a tracking estimate when they are moving. The metrics below are derived per scenario because each scenario tests a different claim.

5.1.2 Metrics

Three families of metric are reported, mirroring the three axes of Section 1.2.

Detection. Per-sample binary classification of room occupancy. A localizer record at time t is treated as a positive prediction when an estimate was emitted ($x \neq \varnothing$) and as a negative prediction otherwise. The ground-truth label at t is determined by the annotation window containing t. This couples the detector and the localizer into a single visible signal: the system is treated as having *declared* occupancy whenever an estimate would be shown to the operator. This is operationally honest, but it conflates the room-level decision $|\mathscr{A}| \geq \theta_L$ (Section 4.4.3) with the localization gate "≥ 2 deduplicated active links yielding a surviving intersection". Both gates draw from the same active-link evaluation; in practice they fire together. The chapter reports per-sample true-positive rate, false-positive rate, and F_1, separately for the frozen and unfrozen configurations.

Localization error. For every sample in an occupied scenario where the localizer emitted an estimate, the Euclidean distance to the ground-truth position at the same instant. For stationary windows the ground-truth point is constant. For moving windows it is interpolated along the polyline using the constant-speed model defined in `eval_localizer.py`. The chapter reports the median, mean, p_{95}, and maximum of this error per pose and pooled across poses.

Coverage. Within each occupied window, the fraction of samples for which an estimate was emitted, $n_{\text{est}}/n_{\text{records}}$. Coverage is reported separately for stationary and moving windows because the two scenarios stress the active-link evidence base in different ways. A complementary metric, the *phantom rate*, is reported for empty windows: the fraction of samples on which an estimate was emitted despite the absence of an occupant. The

phantom rate is numerically the false-positive rate of the detection metric restricted to empty windows.

5.1.3 Aggregation and the constant-speed assumption

Stationary results are reported per pose and pooled. For each pose, the five repetitions are pooled at the per-sample level, so a pose-level statistic is computed on the union of samples from its five trials. Per-trial variability is not reported in the body of the chapter. It is implicit in the spread of the boxplots in Section 5.4.

Moving results inherit a non-trivial ground-truth assumption. The annotation specifies the polyline and the start/end timestamps of the window, and the evaluator interpolates position linearly in arc length. This presumes a constant walking speed, which the experimental protocol approximates but does not enforce: the walker received an audio start cue and a visible countdown sized to the full window, with the path marked at four equally spaced points and a target of fifteen seconds between adjacent markers. Local deviations from constant speed therefore add error to the ground truth, not just to the estimate, and inflate the reported moving error.

5.2 Experimental Setup

The evaluation took place in a single environment over a single subject, with system parameters held fixed. This is a deliberate trade. A narrow setup yields measurements whose differences can be attributed to controlled variables (presence vs. absence, pose, baseline freeze) rather than to confounders (different rooms, different bodies, different parameters). Cross-environment generalization is addressed in Section 5.8 as a limitation rather than as a measurement.

5.2.1 Environment

The room is the *sg-lab* (registered as environment identifier `lab-robotics-ba-eval`), a $7.02\,\mathrm{m} \times 4.44\,\mathrm{m}$ rectangular laboratory at the University of St. Gallen. Ten ESP32-S3 sensing nodes are mounted around the perimeter, 1 m above the ground. Their positions are listed in the room configuration `configs/sg-lab.json` and shown in Figure 5.1. The room contains four tables of standard office dimensions, an unblocked window on the west wall, and two doors. In the top-right corner, the room continues into a small corridor. During every trial, a second person sat motionless in this corridor approximately three meters from the corner. This person is part of the steady-state environment, not a

sensed occupant. The room is served by multiple WiFi infrastructures including *eduroam* and the local lab network. These contribute to the noise floor against which the system must detect motion, and they are part of what makes the setup "realistic" in the sense of Section 1.1.

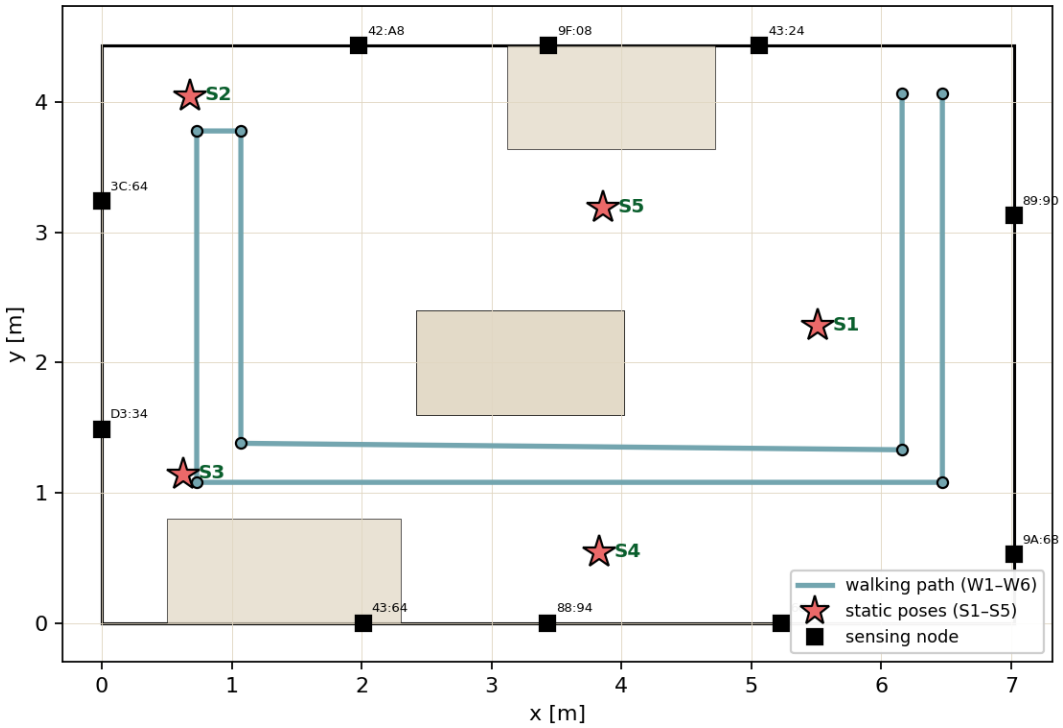

Figure 5.1: The sg-lab evaluation environment (7.02×4.44 m). Ten ESP32-S3 nodes are mounted around the perimeter (black squares, labeled by the last two MAC octets). Five static-occupancy poses S1–S5 (red stars) and the moving polyline W1–W6 (blue) are reused across all trials. Tables are shown in sand-colored fill.

Figure 5.2: Photograph of the evaluation setup.

5.2.2 Subject and Protocol

All occupied scenarios were performed by a single subject, approximately 1.87 m tall and 81 kg. A multi-subject evaluation is outside the scope of this thesis and is named in Section 5.8 as a limitation.

Every trial follows the same four-phase script orchestrated by the experiment runner described in Section 3.9.3: 60 s empty-room baseline, 90 s `occupied_static` at one of the five poses, 60 s `occupied_moving` along the polyline W1–W6, and 60 s empty-room after the subject left. Each phase boundary is signalled by an audio cue. The system is fully recalibrated before every trial, the trial starts as soon as calibration reports ready. This isolates each trial from the drift state of preceding trials and means that the per-trial baseline is built from sixty seconds of empty-room observation that immediately precedes the occupied phases.

5.2.3 System Parameters

All trials use the default detector parameters of `detector.py`: per-link activation threshold $\theta_Z = 2.5$, room-level activation threshold $\theta_L = 3$ active links, EWMA smoothing factor $\alpha = 0.2$, baseline target 2000 samples per link, and a calibration readiness quorum of 70% of viable links. The scheduler uses a slot duration of 20 ms, 10 probe frames per trigger, and a 5 ms inter-slot gap. The localizer uses the parameters of Section 4.5: minimum interior angle 20° and spatial validity margin $\Delta = 0.3$ m.

5.2.4 Trial Inventory

Thirty-one trials were recorded; thirty were usable, the single exclusion being T0050, where an ESP32 node detached from the wall mid-recording, invalidating the link-set assumptions of the localizer for that trial. The exclusion is structural rather than performance-related and is disclosed here for transparency. Twenty-five of them use the *frozen* baseline (Section 3.6.3), with five repetitions at each pose (T0042–T0046 at S1, T0047–T0049, T0051, T0052 at S2, T0053–T0057 at S3, T0058–T0062 at S4, and T0063–T0067 at S5). Five further trials, one per pose, use the *unfrozen* baseline (T0068–T0072).

The unfrozen group is deliberately smaller. The system was originally designed and deployed with the adaptive baseline always active. The frozen-baseline option was added after early trials revealed that an unfrozen baseline absorbs a motionless person within tens of seconds and ceases to detect them (Section 5.6). The five unfrozen trials are sufficient to expose this collapse and to show that, in the moving scenario, the unfrozen configuration behaves similarly to the frozen one. They are not large enough to support per-pose error distributions, and the chapter does not draw any.

5.3 Room-Level Detection

The first question is the simplest one: does the system know whether the room is occupied? Table 5.1 summarises the per-sample confusion of the operational occupancy signal (estimate emitted vs. ground-truth window label) over all trials in each baseline mode.

Table 5.1: Per-sample room-level occupancy classification. n_+ counts samples in occupied windows (stationary $\cup$ moving), n_- samples in empty windows. Prediction is "estimate emitted" on the persisted localizer stream.

Mode	n_+	n_-	TPR	FPR	F_1
frozen	5474	4413	89.8%	1.8%	93.9%
unfrozen	1099	884	28.8%	0.0%	44.8%

Under the frozen baseline, the system declares occupancy correctly on 89.8% of occupied samples and triggers a phantom on only 1.8% of empty samples, yielding an F_1 of 93.9%. This pooled rate is dominated by one trial (T0058); 18 of the 25 frozen trials produced no phantoms at all. The mechanism is discussed in Section 5.7. The

asymmetry between true-positive and false-positive rates is the practically important fact: a missed positive sample costs only a moment of latency because consecutive occupied samples are correlated. A false alarm that triggers an HVAC cycle or a security alert is operationally far more expensive.

Under the unfrozen baseline the picture is qualitatively different. The true-positive rate collapses to 28.8% and the F_1 to 44.8%, while the false-positive rate falls to zero. This is the design's predicted failure: an adaptive baseline absorbs the static signature of a motionless person, and once it has absorbed them, the system reports an empty room. The full-resolution version of this story is told in Section 5.6.

5.4 Stationary Localization (Frozen Baseline)

When the room is correctly declared occupied, how well does the system locate the occupant? For stationary subjects, error decomposes into systematic bias (the estimate consistently displaced from the true pose) and noise (variability of the estimate around its mean). Table 5.2 reports both implicitly through median, mean, p_{95}, and max error per pose; Figure 5.3 shows the full distributions as boxplots.

Table 5.2: Stationary error per pose and pooled, frozen baseline. n is the number of localizer samples on which an estimate was emitted and a Euclidean error to the ground-truth pose could be computed.

Pose	n	Coverage	Median [m]	Mean [m]	p95 [m]	Max [m]
S1	649	98.6%	0.64	0.80	1.74	4.18
S2	451	68.5%	1.36	2.61	6.35	7.52
S3	590	88.9%	0.72	1.68	5.89	7.19
S4	496	76.4%	0.67	1.52	3.64	5.47
S5	623	94.4%	1.00	1.24	2.26	5.01
Pooled	2809	85.4%	0.84	1.50	6.09	7.52

The pooled median is 0.84 m at 85.4% coverage, but the per-pose breakdown is what carries the interpretation. The five static poses, marked on Figure 5.1, sample distinct geometric regimes of the room: two interior poses (S1 in the central-right interior and S5 in the central-upper interior, each at least a metre from any wall), two one-wall-adjacent poses (S3 against the west wall toward its lower end, and S4 against the south wall near its middle), and one extreme-corner pose (S2 wedged into the north-west corner, within a metre of both adjacent walls). Performance tracks this layout directly. The two interior poses are the easy cases: S1 attains 0.64 m median error at 98.6% coverage, and S5

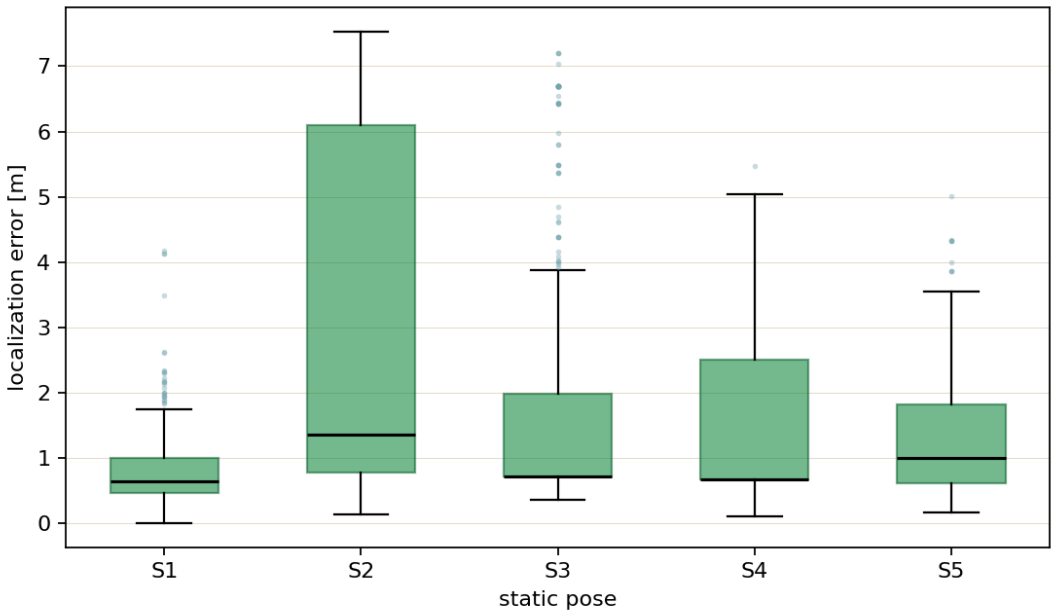

Figure 5.3: Per-pose stationary error distributions (frozen baseline), pooled over the five repetitions at each pose. Box is the inter-quartile range; whiskers extend to 1.5 IQR; individual outliers shown.

follows at 1.00 m and 94.4%. The one-wall-adjacent poses S3 and S4 achieve sub-metre medians (0.72 m and 0.67 m) at intermediate coverage (88.9% and 76.4%). S2 is the conspicuous outlier on every dimension: 1.36 m median, 68.5% coverage, p_{95} of 6.35 m, and a maximum of 7.52 m — a single estimate placed near the opposite corner of the room.

Two observations follow. First, the median is a far better summary of typical performance than the mean or the p_{95}. The pooled mean of 1.50 m is dragged upward by occasional far-displaced estimates whose effect on the geometric centroid is more violent than their frequency. Second, the per-pose spread tracks the line geometry of the link mesh at each pose, not the underlying signal quality. The localizer triangulates by intersecting active link *lines*, where a link's line is the chord between its two endpoints. A pose is accurately localisable only when at least two such chord-lines pass close to the pose *and* cross each other at a non-degenerate angle near it. For interior poses (S1, S5) many cross-wall chords pass through the pose at large angles and the triangulation is well-conditioned. For one-wall-adjacent poses (S3, S4) the relevant chords are those connecting the near wall to nodes on the opposite wall whose along-wall position is close to the pose's, which the deployment happens to provide: at S3 the chords `D3:34`–`43:64`

and D3:34–88:94 pass within 0.1 m of the pose. For S2, no such partnering geometry exists. The only chord whose line passes close to S2 is 3C:64–42:A8. Every other near-S2 link shares an endpoint with 3C:64–42:A8, in which case the crossing is at that shared endpoint. The localizer therefore has effectively one informative line at S2 and cannot triangulate. The detailed enumeration of these candidate chords is given in Section 5.7. Here it is enough to note that the failure is structurally about line coverage of the pose, not signal quality or filter tuning.

Figure 5.4 contextualises the per-pose figures with cumulative distributions. The frozen-stationary curve places 50% of estimates within 0.84 m and 80% within ≈ 2 m, but exhibits a long upper tail extending past 7 m that the median hides.

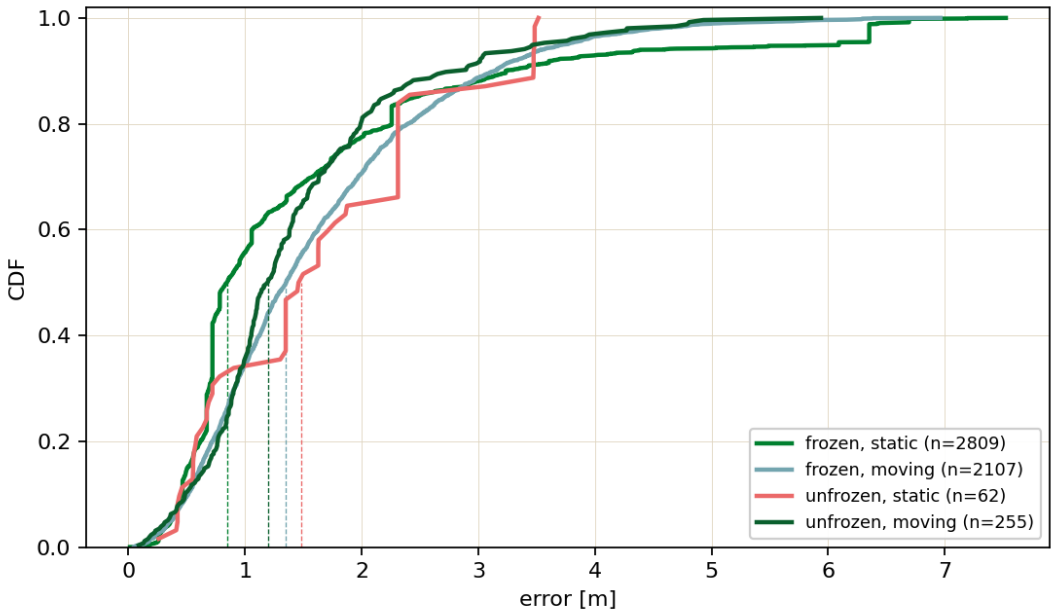

Figure 5.4: Empirical CDFs of localization error across the four mode-by-scenario combinations. Dashed verticals mark each curve's median. The unfrozen-static curve has $n = 62$ because the unfrozen baseline rarely emits an estimate while the subject is still (Section 5.6); its visible shape is dominated by the few moments in which it did.

5.5 Moving Localization

Moving estimates carry the constant-speed ground-truth caveat of Section 5.1.3. The numbers in Table 5.3 should therefore be read as an upper bound on system error rather than as the error itself.

Table 5.3: Moving error per trial group and pooled, frozen baseline. "after S_i" indicates the moving phase of the trial whose static phase was at pose S_i; the walk path itself is identical across trials. n is the number of localizer samples on which both an estimate and an interpolated ground-truth point were available.

Lap context	n	Coverage	Median [m]	Mean [m]	p95 [m]	Max [m]
after S1	416	96.3%	1.38	1.55	3.39	4.89
after S2	426	97.3%	1.29	1.62	3.68	6.38
after S3	428	97.5%	1.44	1.73	3.94	6.96
after S4	417	95.9%	1.43	1.71	3.98	6.38
after S5	420	95.2%	1.22	1.48	3.45	5.94
Pooled	2107	96.4%	1.35	1.62	3.74	6.96

Coverage in the moving scenario is uniformly high ($\geq 95\%$ for every trial group) and the median error is essentially flat at 1.2–1.4 m across all five trial groups. As expected, moving consistently produces more active links than stationary because the occupant disturbs more of the multipath structure per unit time and because the EWMA does not have time to relax onto a single quasi-stable state. Coverage is therefore higher than in the stationary case while the median error is slightly worse, likely because the localizer is producing a position estimate against a moving target whose interpolated ground truth has its own error.

Figure 5.5 illustrates the shape of a single trial. The empty phases (yellow shading) carry no estimates and therefore no error points. The static phase (green) shows a tight band of error around the pose median of approximately 0.6–1.0 m, with a few brief excursions to 1.5 m. The moving phase (blue) shows the wider error band typical of motion: median around 1.3 m with some excursions to 4 m.

5.6 Baseline Freeze: Frozen vs. Unfrozen

The most consequential single design choice in the system, visible to the operator as a toggle in the monitoring interface (Section 3.10), and structurally motivated in the design chapter (Section 3.6.3), is whether the Welford baseline keeps adapting once the system has entered monitoring mode. The design predicts that an adaptive baseline absorbs a motionless person within tens of seconds. The frozen-vs-unfrozen comparison tests this prediction directly.

Table 5.4 pairs the per-pose stationary and moving statistics under the two configurations. The contrast is stark. Frozen static coverage averages 85% and never drops

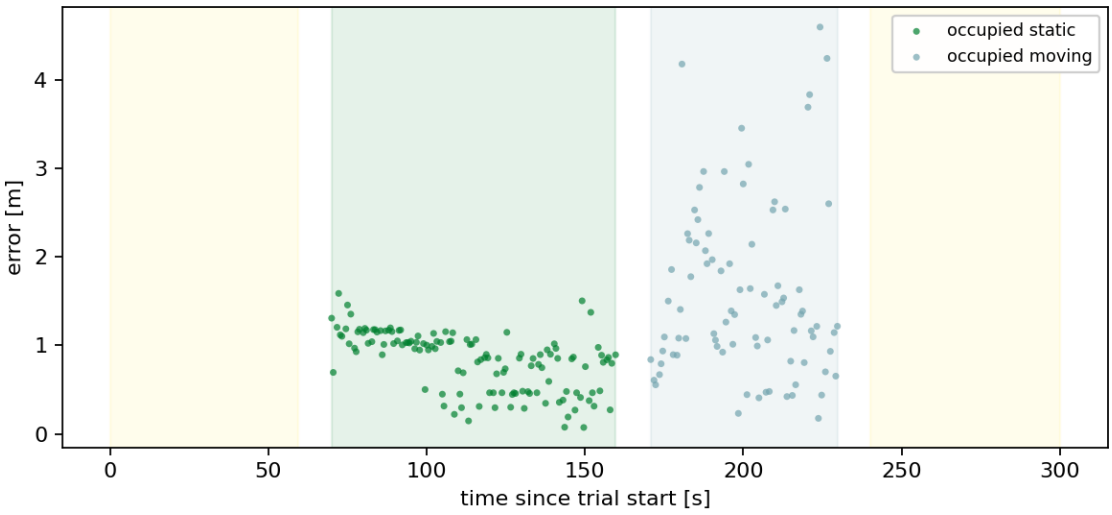

Figure 5.5: Single-trial error timeline for T0042 (frozen baseline, static pose S1). Phase shading is yellow for empty, green for stationary, blue for moving. The empty phases produce no estimates. The static phase tracks tightly around the median for S1. The moving phase shows the wider band with some excursions to 4 m.

below 68%. Unfrozen static coverage averages 9.4% and reaches zero at pose S2. The moving coverage figures differ less dramatically — frozen 96%, unfrozen 59% — and the moving median error is essentially the same in both modes (1.3 m vs. 1.2 m). The fact that the unfrozen baseline's moving performance survives at all confirms that the unfrozen failure is specific to stationarity, not a general detection failure. When the occupant keeps moving, fresh variance arrives faster than the baseline can absorb it, and the system continues to locate them.

Figure 5.6 shows the time-resolved version of the same finding. Each 90 s static window is divided into eighteen 5 s bins. For each bin, the fraction of samples on which an estimate was emitted is plotted, pooled across all five poses. The frozen curve fluctuates around 85% throughout the window. The unfrozen curve tells the predicted story: it starts at roughly the same level as the frozen curve (the baseline has not yet had time to adapt), decays through 20 seconds, and reaches zero coverage by approximately 25 seconds into the window. After that point the system, in the unfrozen configuration, has effectively forgotten that the room is occupied. The small late-window bump around 40–70 seconds was mostly caused by T0068 (see B.1) and likely stems from the subject adjusting its posture. This could have re-triggered the localizer for a few samples and then decay again.

Table 5.4: Frozen vs. unfrozen baseline, per pose. "Static cov." is the fraction of the 90 s static window on which an estimate was emitted. "Static median" is the median error over those samples, with "–" indicating that no samples were available because the baseline absorbed the occupant entirely.

Pose	Mode	Static cov.	Static median [m]	Moving cov.	Moving median [m]
S1	frozen	98.6%	0.64	96.3%	1.38
S1	unfrozen	25.8%	1.87	42.0%	1.09
S2	frozen	68.5%	1.36	97.3%	1.29
S2	unfrozen	0.0%	–	67.8%	1.14
S3	frozen	88.9%	0.72	97.5%	1.44
S3	unfrozen	5.3%	3.06	59.3%	1.29
S4	frozen	76.4%	0.67	95.9%	1.43
S4	unfrozen	6.7%	0.56	60.9%	1.13
S5	frozen	94.4%	1.00	95.2%	1.22
S5	unfrozen	9.1%	1.35	63.2%	1.38

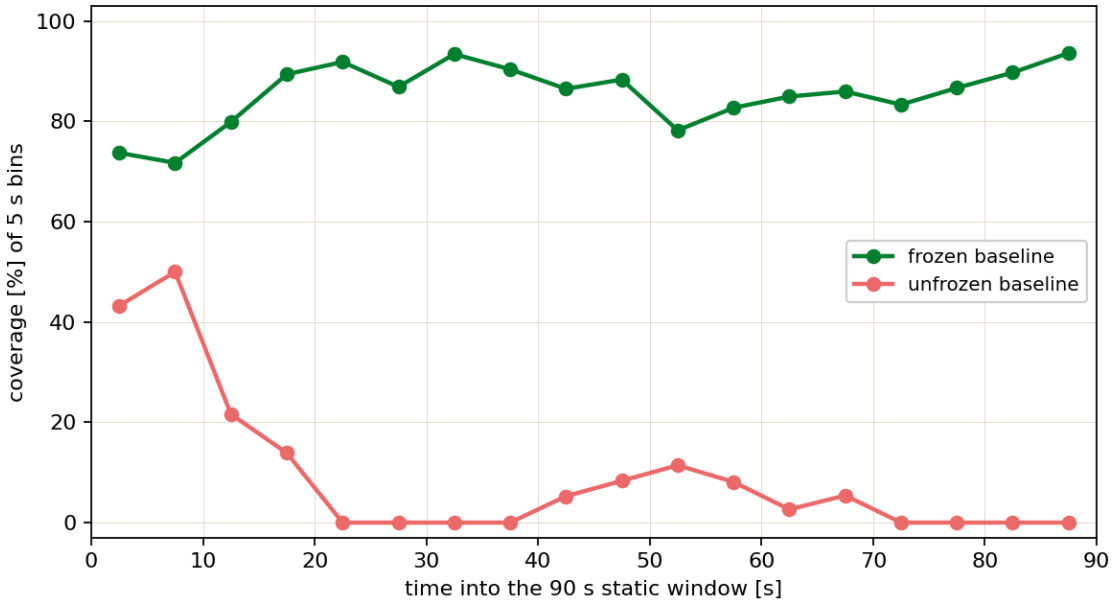

Figure 5.6: Coverage as a function of time into the 90 s static window, pooled over all five poses. The frozen baseline (green) sustains coverage above 75% for the entire window; the unfrozen baseline (red) decays to zero coverage by approximately 25 s.

The picture is consistent across all five poses individually (the per-pose decay traces are pooled in the figure but the per-pose entries of Table 5.4 confirm the pattern: 0% coverage at S2, 5–10% at S3/S4/S5, 25.8% at S1). The fact that no pose escapes the collapse is what makes the freeze toggle structural rather than environmental. The freeze is not a tuning convenience. Without it, the system does not detect stationary occupants at all.

5.7 Failure Modes

The aggregate numbers in the preceding sections conceal a set of distinct failure modes whose mechanisms are worth naming individually, because each suggests a different remediation.

Geometric undercoverage at S2. The clearest single failure pattern in the stationary data is pose S2 (Figure 5.3, Table 5.2): a wide error distribution, a p_{95} of 6.35 m, and a maximum estimate placed near the opposite corner of the room. The mechanism is described in Section 5.4: only one link line in the present mesh passes near S2. Every other near-S2 link line shares an endpoint with the nearest one, so even if two of them activate, they do not yield any useful triangulation.

This is a structural property of the mesh layout, not a tuning issue. No setting of θ_Z, θ_L, or the spatial-filter tolerance can manufacture link geometry that is not there to begin with. The remediation must therefore happen at deployment time, by ensuring that every region the system is expected to localise within is crossed by at least two link lines drawn from *distinct node pairs* (so that pairwise intersections are not pinned to a shared endpoint, as happens at S3 and S4 — Section 5.4).

For the present room the practical fix is to add a single node on the same walls that frame the affected corner, but closer to the corner itself: a node at roughly $(0.5, 4.44)$ on the north wall would form links with the south-wall nodes 43:64, 88:94, and 6C:64 whose lines would pass 0.04 m, 0.06 m, and 0.16 m from S2 respectively, each through a different partner, supplying the diversity of intersection geometry the corner currently lacks.

Such regions can be spotted at a glance in the monitoring interface(B.2), which overlays the link lines on the room map. The wide reported radii at S2 (Figure 5.7) reflect the same weakness numerically, as the explicit null output (Section 3.1) intended.

Phantom estimates and post-occupancy baseline drift. The pooled frozen-baseline phantom rate of 1.8% is highly asymmetric in time. In the empty window before

occupancy, when the frozen baseline was captured, only 5 phantoms occur in 2228 samples (0.22%). In the empty window after occupancy, the rate rises to 3.39% (74 phantoms in 2185 samples), a fifteen-fold increase. The asymmetry is consistent across trials, and one trial (T0058) reaches the limiting case: 60 phantoms over the full 60 s post-occupancy window, distributed uniformly with a median inter-arrival gap of 0.55 s. The mechanism is structural rather than transient. The frozen baseline is, by construction, a snapshot of the room state at a single moment before the occupant enters. By the time the occupant leaves, the multipath structure has shifted slightly — a chair displaced, a door at a different angle, the settled multipath around the now-empty body still relaxing — and the frozen reference no longer describes the actual empty state. On most trials the residual shift is small enough that the two-level θ_Z, θ_L gate still suppresses it. On T0058 it is not. This failure mode is the price the freeze design pays for not letting an adaptive baseline absorb the occupant.

Radius miscalibration on geometrically weak intersections. The localizer's reported radius (Section 4.5.5) is intended as an honest uncertainty estimate that the operator can read off the live display. Figure 5.7 shows the reported radius against the realised error. The relationship is loose. A perfectly calibrated radius would track the $y = x$ diagonal. The observed distribution sits mostly below it, indicating that the radius systematically over-estimates the realised error when the intersection cloud is diffuse, and under-estimates it (a vertical band near $r \approx 1$ m) when a single-intersection fallback is used. The radius is therefore directionally informative: a large reported radius corresponds to larger error on average, but it is not quantitatively calibrated.

Unfrozen-baseline absorption. The dominant failure mode under the unfrozen baseline is the one for which the entire freeze mechanism exists. It is treated separately in Section 5.6 and further discussed in Section 5.8.5.

5.8 Discussion

This section relates the empirical results back to the five objectives stated in Section 1.2, addressing each in turn before stepping back to the research question itself.

5.8.1 Adapting the inherited sensing pipeline

The first objective was to reshape the inherited pipeline so that it learns its reference from the empty room and translates per-link channel deviations into a room-level occupancy decision and a two-dimensional position estimate with associated uncertainty. The

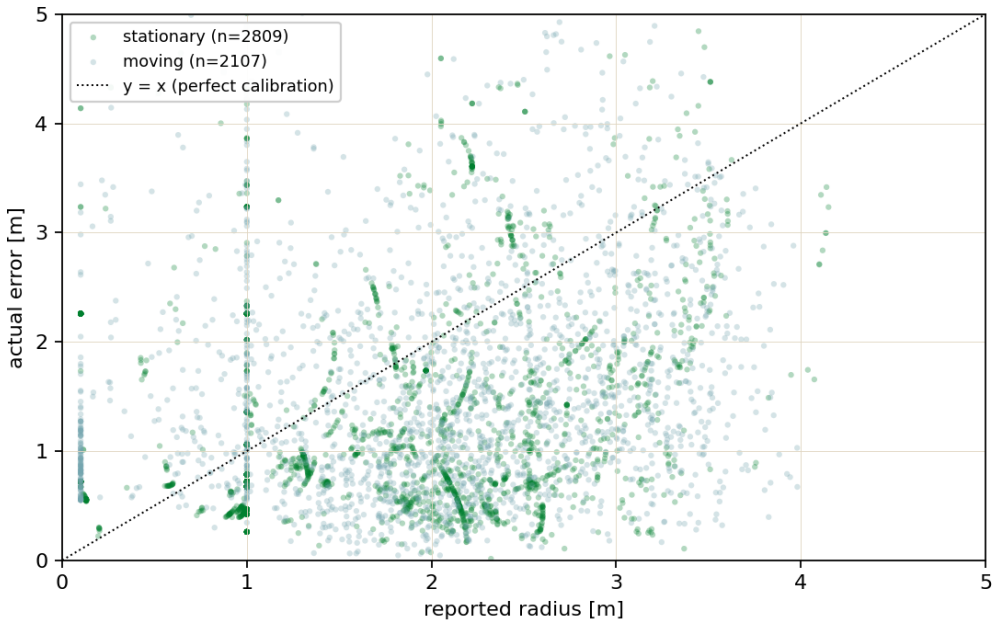

Figure 5.7: Reported localizer radius against realised error, pooled across all frozen-baseline trials. Each point is one localizer sample. The dotted $y = x$ line marks the perfect-calibration ideal: actual error equal to reported radius. The vertical band at $r = 1\,\text{m}$ is the single-intersection fallback (Section 4.5.5).

evaluation confirms that this objective is met. The system produces all three outputs as specified: a room-level decision (Section 5.3), a position estimate (Sections 5.4 and 5.5), and an uncertainty radius (Figure 5.7). The reference itself is learned online during the calibration phase (approximately 60 to 90 seconds). No labeled occupant data is consumed at any point. The unfrozen-baseline experiments (Section 5.6) additionally demonstrate why the freeze mechanism, added during development, is a structural rather than cosmetic part of the design: without it, the system meets the objective for moving occupants but fails it for stationary ones.

5.8.2 Deploying on ESP32-S3 commodity hardware.

The second objective was a reproducible deployment on commodity nodes. Thirty usable trials were recorded on a ten-node ESP32-S3 mesh running identical firmware, with the only mid-experiment hardware event being the single node that physically detached from the wall during T0050. Throughout these trials the observed per-link sampling rate was 10–50 Hz, well above the viability floor of 0.5 Hz, and the backend was not throughput-limited. The deployment methodology described in Section 4.1 therefore

produces a system that runs stably for thirty consecutive trial sessions on commodity hardware. This is a stronger statement than the design chapter alone could justify.

5.8.3 Empirical evaluation along detection, localization error, and coverage.

The third objective was the production of the evidence base summarised in Sections 5.3 to 5.6. Under the frozen baseline the system achieves 89.8% per-sample TPR at1.8% FPR ($F_1 = 93.9\%$), median stationary error of 0.84 m at 85.4% coverage, and median moving error of 1.35 m at 96.4% coverage. These figures are coarser than the best reported numbers in the literature — FILA achieves ≈ 0.5 m median error with an Intel 5300 NIC [35], LiFS reaches comparable accuracy with eleven laptops in line-of-sight conditions [31], SpotFi attains ≈ 0.4 m median error using joint AoA–ToF super-resolution across multiple access points [17], and Dang et al. place $\approx 87\%$ of test positions within 1 m [6] — but each of those systems either depends on Intel 5300-class hardware with full CSI access, requires an offline fingerprinting phase, or relies on per-deployment calibration of phase super-resolution. The closest comparator on commodity hardware without offline training is WIDE, which reports ≈ 0.95 m average tracking error in a 3×3 transceiver grid [10]; the present system operates in the same regime while additionally producing an uncertainty radius, and without WIDE's requirement of a known starting location and continuous motion. The results are therefore favorable for an operationally useful occupancy detector and a coarse localizer in a room of 7×4.4 m served by ten commodity ESP32-S3 nodes, since the system did not consume labeled training data at the time of deployment.

5.8.4 Identifying and characterising failure modes.

The fourth objective specified that the evaluation produce concrete starting points for future improvements rather than aggregate metrics alone. Section 5.7 enumerates four such starting points. The S2 weakness is addressable by adding one node on the walls that frame the affected corner, closer to the corner than any existing node, so that its links to nodes on the opposite walls pass within Fresnel-zone distance of the pose and supply the intersection diversity the mesh currently lacks there.

The phantom rate exhibits a systematic post-occupancy asymmetry: it is roughly fifteenfold higher in the empty window after occupancy than in the empty window before it and is concentrated in a single trial where the system fired for the full minute after the subject left. This shows a permanently frozen baseline is unlikely to remain valid

in practice. Even within the brief span of a single trial, the room's multipath structure drifts enough that the original reference no longer represents the current empty state. Any longer-running deployment would compound this with the slower environmental changes (furniture displacement, equipment, door positions, other occupants in adjacent spaces) that the original calibration cannot anticipate.

The radius miscalibration is addressable by replacing the single-intersection fallback with an explicit "low-confidence" flag.

5.8.5 Gap between research prototype and practical deployment.

The fifth objective was the distillation of the evidence into deployment-relevant recommendations. Three are warranted by the data above.

Node Placement and Link-Line Coverage. Node placement determines per-pose error through a single geometric criterion: every location the system is expected to localise within should be crossed by at least two link *lines* (segments between pairs of nodes) drawn from distinct node pairs, with both lines passing inside the Fresnel-zone radius of the link at that location. A deployment should run this audit against its node plan before installation, treating it as a binding constraint rather than a target. Performing the audit is impractical by hand on non-trivial rooms. More usefully, searching for layouts that pass the audit is also impractical to do manually. A concrete next step is to develop a placement-optimization tool: given a room outline, a target node count, and the mounting constraints of the building (wall surfaces, mounting heights, power availability), the tool searches for layouts that maximise the room area satisfying the criterion.

Bounded-Weight Baseline Adaptation. Neither of the two baseline regimes tested here is the right long-term answer. The unfrozen baseline absorbs a motionless person within $\approx 25\,\mathrm{s}$ (Section 5.6); the permanently frozen baseline drifts out of validity within a single trial, manifesting as the fifteenfold post-occupancy phantom asymmetry of Section 5.7. The mechanism behind the unfrozen failure is visible directly in the Welford update used by the baseline (Section 4.3.1): each new sample contributes a weight of $1/n$ to the running mean, where n is the number of samples accumulated so far. Calibration terminates at the target of 2000 samples per link, and the 60 s empty-room baseline phase that follows (Section 5.2.2), at the observed per-link rate of 40–50 Hz, extends n to roughly 4000–5000 samples per link by the time the subject enters. Each new sample at that point therefore carries a weight of $\sim 0.02\%$ in the running mean. Over the $\approx 25\,\mathrm{s}$ absorption window, on the order of 1000 consecutive samples drawn from the

static signature accumulate enough weight to drag the running mean toward it, pulling the per-link Z-score below the activation threshold θ_Z. If the system were left running indefinitely without intervention, n would keep growing and each subsequent sample's weight would shrink toward zero, so the baseline would asymptotically behave like a frozen one: it would track slow drift early on, then freeze in place after enough samples accumulated.

A simple modification could potentially resolve both issues. Rather than letting n continue to grow during monitoring, n is fixed at a chosen value $n^\star$ the moment monitoring begins and held there: every subsequent Welford update uses $\mu \mathrel{+}= \delta/n^\star$, so the per-sample weight is pinned at $1/n^\star$ indefinitely. The value of $n^\star$ should be chosen so that this fixed weight is small enough that the cumulative contribution of a stationary occupant over the time it might plausibly remain still does not displace the baseline past the per-link Z-score threshold θ_Z, but large enough that the baseline can still follow environmental drift on the minute-to-hour timescale at which chairs are moved, doors swing, and multipath relaxes. The empirical timescales in Section 5.6 (≈ 25 s to absorb a static occupant at the present operating point) and Section 5.7 (drift visible within a single ~ 4 minute trial) bracket the range $n^\star$ would need to occupy; choosing it precisely is a tuning exercise for a future deployment rather than a result of this evaluation.

Single-Intersection Fallback The single-intersection fallback currently assigns a fixed radius of ≈ 1 m regardless of the realised error (the vertical band in Figure 5.7). Presenting this on the same numerical scale as the multi-intersection radii invites the operator to read a precision into the estimate that the geometry does not support. Two remedies are open. Either the fallback radius is raised to a value that bounds the realised error in this regime, making the reported number an honest metric upper bound rather than an artefact of the fallback path, or the fallback estimate is labeled with a low-confidence flag instead of a metric distance. Both are software changes confined to the localizer's output stage.

5.8.6 Threats to validity.

Our evaluation is subject to several constraints that limit external validity.

- **Single subject and environment:** All trials involve a single occupant in a single room, precluding evaluation of multi-occupant interactions and cross-room generalization.

- **Temporal reset assumption:** Each experiment begins from fresh calibration, so long-term drift under stale empty-room references is not characterized.
- **Static environment:** The room layout is fixed, excluding furniture or structural changes that would invalidate the learned baseline.
- **Ground-truth simplification:** Moving-error metrics assume constant-speed interpolation (Section 5.1.3), introducing an unquantified bias.

These limitations reflect experimental scope rather than methodological failure.

Chapter 6

Conclusion

This thesis asked what the practical capabilities and limitations of WiFi CSI-based occupancy detection and localization are on commodity, low-cost hardware in a realistic indoor environment, when the system operates without per-site labeled training data. The short answer the evidence in Chapter 5 supports is that such a system is achievable: within a single calibrated room of $7 \times 4.4\,\text{m}$ served by ten ESP32-S3 nodes, it functions as a usable occupancy detector ($F_1 = 93.9\%$) and a coarse localizer (median error $0.84\,\text{m}$ static, $1.35\,\text{m}$ moving), with no labeled occupant data consumed at any point. Its limitations are dominated not by the underlying sensing principle but by two structural concerns — the geometry of the node mesh and the time horizon over which the empty-room reference remains valid — both of which the evaluation isolates and which the recommendations below address directly.

6.1 Summary

Chapter 2 establishes the technical and scholarly context: how WiFi packets give rise to a per-subcarrier channel estimate, why that estimate is sensitive to nearby bodies, and which signal-processing steps the literature builds on top of it. The accompanying survey identifies the gap this thesis targets — the field has accumulated promising laboratory demonstrations but under-characterized the conditions under which such systems work on commodity hardware without per-site labeled data.

Chapter 3 develops the system. An inherited pipeline is reshaped into a centrally coordinated, TDMA-scheduled mesh whose CSI is reduced to a per-link variance feature, smoothed by an EWMA, and compared against an online Welford baseline learned from the empty room. A two-stage gate (θ_Z per link, θ_L at the room level) yields the occupancy decision; a geometry-based localizer intersects active-link line segments and emits a weighted-centroid position with a confidence radius, or an explicit null when the evidence

is insufficient. The baseline-freeze toggle is included as a deliberate design choice, motivated by the absorption pathology of any purely adaptive baseline.

Chapter 4 realises this design. Ten ESP32-S3 nodes run a single firmware image and communicate with a centralized backend over a compact binary protocol. The backend implements the calibration pipeline (Welford accumulators, link-viability heuristics, readiness quorum, adaptive slot allocation), link activation, the localizer, and a live monitoring interface used for both operation and diagnostic inspection of link geometry. The combined system was brought up on commodity hardware and ran for thirty consecutive recording sessions without intervention.

Chapter 5 characterises that system empirically. Thirty usable trials in the *sg-lab* environment follow a fixed four-phase script (empty / static at one of five poses S1–S5 / moving along a fixed polyline / empty), under both frozen and unfrozen baselines. Detection, per-pose stationary error, moving error, and coverage are reported with their full distributions; the freeze toggle is studied as its own controlled comparison; failure modes are named individually and traced to their mechanisms.

6.2 Lessons Learned

Four findings are worth carrying away from this work.

First, the system is operationally useful in its present form. Under the frozen baseline it correctly declares occupancy on 89.8% of occupied samples at a 1.8% phantom rate, with median stationary error 0.84 m at 85% coverage and median moving error 1.35 m at 96% coverage.

Second, the baseline-freeze toggle is structural, not cosmetic. Without it, the adaptive Welford baseline absorbs a motionless occupant within approximately 25 s and coverage collapses to zero. With it, stationary coverage is sustained throughout a 90 s window. The toggle therefore separates the regime in which the system locates a still occupant from the regime in which it does not.

Third, a permanently frozen baseline is not the long-term answer. The phantom rate after occupancy is roughly fifteen times the rate before occupancy, a systematic asymmetry that reveals the frozen reference drifting out of validity within the span of a single trial. Neither of the two regimes tested in the chapter is sustainable as deployed; the right mechanism lies between them.

Fourth, per-pose error is set by mesh geometry rather than by signal quality or filter tuning. Interior poses (S1, S5) and one-wall-adjacent poses (S3, S4) yield sub-metre

medians; the corner pose S2, for which only a single link line passes near the pose, exhibits a p_{95} of 6.35 m. No setting of θ_Z, θ_L, or the spatial-filter tolerance can manufacture link geometry that is not there to begin with. Where the lines do not cross near a region, the system cannot locate within it.

6.3 Future Work

The recommendations developed in Section 5.8.5 carry over directly and are restated here as the most concrete next steps.

A placement-optimisation tool. The per-pose results show that deployment-time mesh geometry sets the floor on localisation error. A practical next step is to develop a tool that, given a room outline, a target node count, and the building's mounting constraints, searches for layouts in which every location of interest is crossed by at least two link lines drawn from distinct node pairs and within Fresnel-zone distance of the location. The auditing criterion is already specified; what is missing is the search procedure that makes the criterion usable on non-trivial rooms.

Bounded-weight baseline adaptation. The most defensible long-term replacement for the binary freeze toggle is a Welford update in which the per-sample weight is pinned at a fixed $1/n^\star$ during monitoring rather than allowed to keep shrinking as samples accumulate. The empirical timescales of Section 5.6 (≈ 25 s to absorb a static occupant at the current operating point) and Section 5.7 (drift visible within a single trial) bracket the range that $n^\star$ would need to occupy. Choosing it precisely is a tuning exercise that a longer-running deployment can resolve.

An honest single-intersection fallback. The localizer's single-intersection fallback currently emits a fixed radius of approximately 1 m regardless of the realised error, presenting an artificial precision on the live display. Replacing the fixed radius with an explicit low-confidence flag, or with a radius value that bounds the realised error in this regime, is a software change confined to the localizer's output stage and would close the gap between the reported uncertainty and the geometry that supports it.

Taken together, these three directions follow the same thread as the evaluation itself: that the system's residual limitations are addressable structural choices — placement geometry, baseline dynamics, and honest uncertainty — rather than fundamental limits of CSI sensing on commodity hardware. The principal contribution of this thesis is the evidence that supports that distinction.

Bibliography

[1] Chittaranjan Andrade. "Z scores, standard scores, and composite test scores explained". In: *Indian Journal of Psychological Medicine* 43.6 (2021), pp. 555–557. DOI: 10.1177/02537176211046525.

[2] Andrea Brunello, Angelo Montanari, Raúl Montoliu, Adriano Moreira, Nicola Saccomanno, Emilio Sansano-Sansano, and Joaquín Torres-Sospedra. "Time matters: Empirical insights into the limits and challenges of temporal generalization in CSI-based Wi-Fi sensing". In: *Internet of Things* 32 (2025), p. 101634. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2025.101634.

[3] Marco Cominelli, Francesco Gringoli, and Francesco Restuccia. "Exposing the CSI: A Systematic Investigation of CSI-based Wi-Fi Sensing Capabilities and Limitations". In: *2023 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. 2023, pp. 81–90. DOI: 10.1109/PERCOM56429.2023.10099368.

[4] *Concepts of orthogonal Frequency Division multiplexing (OFDM) and 802.11 WLAN*. URL: https://helpfiles.keysight.com/csg/89600B/Webhelp/Subsystems/wlan-ofdm/content/ofdm_basicprinciplesoverview.htm.

[5] *CSI Sanitization | Hands-on Wireless Sensing with Wi-Fi: A Tutorial*. Tsinghua University. URL: https://tns.thss.tsinghua.edu.cn/wst/docs/sanitization (visited on 04/21/2026).

[6] Xiaochao Dang, Xuhao Tang, Zhanjun Hao, and Yang Liu. "A Device-Free Indoor Localization Method Using CSI with Wi-Fi Signals". In: *Sensors* 19.14 (2019). ISSN: 1424-8220. DOI: 10.3390/s19143233.

[7] Guillermo Diaz, Iker Sobron, Iñaki Eizmendi, Iratxe Landa, Johana Coyote, and Manuel Velez. "Channel phase processing in wireless networks for human activity recognition". In: *Internet of Things* 24 (2023), p. 100960. ISSN: 2542-6605. DOI: 10.1016/j.iot.2023.100960.

[8] *ESP-IDF Programming Guide: Wi-Fi Driver*. Version 5.2.3. Espressif Systems.

[9] *ESP32-S3 Series Datasheet*. Version 2.2. Espressif Systems.

[10] Jian Fang, Lei Wang, Zhenquan Qin, Bingxian Lu, Wenbo Zhao, Yixuan Hou, and Jenhui Chen. "A Lightweight Passive Human Tracking Method Using Wi-Fi". In: *Sensors* 22.2 (2022). ISSN: 1424-8220. DOI: 10.3390/s22020541.

[11] Xu Feng, Khuong An Nguyen, and Zhiyuan Luo. "A survey of deep learning approaches for WiFi-based indoor positioning". In: *Journal of Information and Telecommunication* 6 (2021), pp. 1–54. DOI: 10.1080/24751839.2021.1975425.

[12] Yanying Gu, Anthony Lo, and Ignas Niemegeers. "A survey of indoor positioning systems for wireless personal networks". In: *IEEE Communications Surveys Tutorials* 11.1 (2009), pp. 13–32. DOI: 10.1109/SURV.2009.090103.

[13] Linqing Gui, Wenyang Yuan, and Fu Xiao. "CSI-based passive intrusion detection bound estimation in indoor NLoS scenario". In: *Fundamental Research* 3.6 (2023), pp. 988–996. ISSN: 2667-3258. DOI: https://doi.org/10.1016/j.fmre.2022.05.015.

[14] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. "Tool Release: Gathering 802.11n Traces with Channel State Information". In: *Computer Communication Review* 41 (2011), p. 53. DOI: 10.1145/1925861.1925870.

[15] Steven M. Hernandez and Eyuphan Bulut. "Lightweight and Standalone IoT Based WiFi Sensing for Active Repositioning and Mobility". In: *2020 IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. 2020, pp. 277–286. DOI: 10.1109/WoWMoM49955.2020.00056.

[16] Karim Khamaisi and Bruno Rodrigues. *Less is More: The Dilution Effect in Multi-Link Wireless Sensing*. 2026. arXiv: 2602.10823 [cs.NI]. URL: https://arxiv.org/abs/2602.10823.

[17] Manikanta Kotaru, Kiran Joshi, Dinesh Bharadia, and Sachin Katti. "SpotFi: Decimeter Level Localization Using WiFi". In: *SIGCOMM Comput. Commun. Rev.* 45.4 (2015), pp. 269–282. ISSN: 0146-4833. DOI: 10.1145/2829988.2787487.

[18] Il-Gu Lee and Sok-Kyu Lee. "Efficient automatic gain control algorithm and architecture for wireless LAN receivers". In: *Journal of Systems Architecture* 53.7 (2007), pp. 379–385. ISSN: 1383-7621. DOI: https://doi.org/10.1016/j.sysarc.2006.11.002.

[19] Junshuo Liu, Fuhai Wang, Zhe Li, Rujing Xiong, Tiebin Mi, and Robert Caiming Qiu. *A Wi-Fi Signal-Based Human Activity Recognition Using High-Dimensional Factor Models*. 2023. arXiv: 2311.05921 [eess.SY]. URL: https://arxiv.org/abs/2311.05921.

[20] Yongsen Ma, Gang Zhou, and Shuangquan Wang. "WiFi Sensing with Channel State Information: A Survey". In: *ACM Comput. Surv.* 52.3 (2019). ISSN: 0360-0300. DOI: 10.1145/3310194.

[21] Miguel Matey-Sanz, Joaquín Torres-Sospedra, and Adriano Moreira. "Temporal Stability on Human Activity Recognition based on Wi-Fi CSI". In: *2023 13th International Conference on Indoor Positioning and Indoor Navigation (IPIN)*. 2023, pp. 1–6. DOI: 10.1109/IPIN57070.2023.10332214.

[22] Marcus Perry. "The Exponentially Weighted Moving Average". In: 2010. ISBN: 9780470400630. DOI: 10.1002/9780470400531.eorms0314.

[23] Lin Qi, Yu Liu, Yue Yu, Liang Chen, and Ruizhi Chen. "Current Status and Future Trends of Meter-Level Indoor Positioning Technology: A Review". In: *Remote Sensing* 16.2 (2024). ISSN: 2072-4292. DOI: 10.3390/rs16020398.

[24] Vishnu V. Ratnam, Hao Chen, Hao-Hsuan Chang, Abhishek Sehgal, and Jianzhong Zhang. "Optimal Preprocessing of WiFi CSI for Sensing Applications". In: *IEEE Transactions on Wireless Communications* 23.9 (2024), pp. 10820–10833. DOI: 10.1109/TWC.2024.3376332.

[25] Charles E. Spurgeon. *Ethernet: The Definitive Guide*. 1st ed. 2000. ISBN: 1-56592-660-9.

[26] Sheng Tan, Yili Ren, Jie Yang, and Yingying Chen. "Commodity WiFi Sensing in Ten Years: Status, Challenges, and Opportunities". In: *IEEE Internet of Things Journal* 9.18 (2022), pp. 17832–17843. DOI: 10.1109/JIOT.2022.3164569.

[27] Mario Roos-Hoefgeest Toribio, Alejandro Garnung Menéndez, Sara Roos-Hoefgeest Toribio, and Ignacio Álvarez García. "A novel approach to speed up hampel filter for outlier detection". In: *Sensors* 25.11 (2025), p. 3319. DOI: 10.3390/s25113319.

[28] Marek Wadinger and Michal Kvasnica. *Adaptable and Interpretable Framework for Novelty Detection in Real-Time IoT Systems*. 2023. arXiv: 2304.02947 [cs.LG]. URL: https://arxiv.org/abs/2304.02947.

[29] Fei Wang, Tingting Zhang, Wei Xi, Han Ding, Ge Wang, Di Zhang, Yuanhao Cui, Fan Liu, Jinsong Han, Jie Xu, and Tony Xiao Han. "A Survey on Wi-Fi Sensing Generalizability: Taxonomy, Techniques, Datasets, and Future Research Prospects". In: *IEEE Communications Surveys amp; Tutorials* 28 (2026), pp. 5227–5266. ISSN: 2373-745X. DOI: 10.1109/comst.2026.3670854.

[30] Hao Wang, Daqing Zhang, Junyi Ma, Yasha Wang, Yuxiang Wang, Dan Wu, Tao Gu, and Bing Xie. "Human respiration detection with commodity wifi devices: do user location and body orientation matter?" In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 2016, pp. 25–36. ISBN: 9781450344616. DOI: 10.1145/2971648.2971744.

[31] Ju Wang, Hongbo Jiang, Jie Xiong, Kyle Jamieson, Xiaojiang Chen, Dingyi Fang, and Binbin Xie. "LiFS: low human-effort, device-free localization with fine-grained subcarrier information". In: *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. 2016, pp. 243–256. ISBN: 9781450342261. DOI: 10.1145/2973750.2973776.

[32] Kailong Wang, Cong Shi, Jerry Cheng, Yan Wang, Minge Xie, and Yingying Chen. "Solving the WiFi Sensing Dilemma in Reality Leveraging Conformal Prediction". In: *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*. 2023, pp. 407–420. ISBN: 9781450398862. DOI: 10.1145/3560905.3568529.

[33] Zhengjie Wang, Zehua Huang, Chengming Zhang, Wenwen Dou, Yinjing Guo, and Da Chen. "CSI-based human sensing using model-based approaches: a survey". In: *Journal of Computational Design and Engineering* 8.2 (2021), pp. 510–523. ISSN: 2288-5048. DOI: 10.1093/jcde/qwab003.

[34] Bo Wei, Hang Song, Jiro Katto, and Takamaro Kikkawa. *RSSI-CSI Measurement and Variation Mitigation with Commodity WiFi Device*. 2022. arXiv: 2203.12888 [eess.SP]. URL: https://arxiv.org/abs/2203.12888.

[35] Kaishun Wu, Jiang Xiao, Youwen Yi, Min Gao, and Lionel M. Ni. "FILA: Fine-grained indoor localization". In: *2012 Proceedings IEEE INFOCOM*. 2012, pp. 2210–2218. DOI: 10.1109/INFCOM.2012.6195606.

[36] Jiang Xiao, Huichuwu Li, Minrui Wu, Hai Jin, M. Jamal Deen, and Jiannong Cao. "A Survey on Wireless Device-free Human Sensing: Application Scenarios, Current Solutions, and Open Issues". In: *ACM Comput. Surv.* 55.5 (2022). ISSN: 0360-0300. DOI: 10.1145/3530682.

[37] Xuecheng Xie, Dongheng Zhang, Yadong Li, Yang Hu, Qibin Sun, and Yan Chen. "Robust WiFi Respiration Sensing in the Presence of Interfering Individual". In: *IEEE Transactions on Mobile Computing* 23.8 (2024), pp. 8447–8462. DOI: 10.1109/TMC.2023.3348879.

[38] Zheng Yang, Zimu Zhou, and Yunhao Liu. "From RSSI to CSI: Indoor localization via channel response". In: *ACM Comput. Surv.* 46.2 (2013). ISSN: 0360-0300. DOI: 10.1145/2543581.2543592.

[39] Daqing Zhang, Youwei Zeng, Fusang Zhang, and Jie Xiong. "Chapter 11 - WiFi CSI-based vital signs monitoring". In: *Contactless Vital Signs Monitoring*. 2022, pp. 231–255. ISBN: 978-0-12-822281-2. DOI: https://doi.org/10.1016/B978-0-12-822281-2.00020-2.

[40] Guozhen Zhu, Yuqian Hu, Chenshu Wu, Wei-Hsiang Wang, Beibei Wang, and K. J. Ray Liu. *Experience Paper: Scaling WiFi Sensing to Millions of Commodity Devices for Ubiquitous Home Monitoring*. 2025. arXiv: 2506.04322 [eess.SP]. URL: https://arxiv.org/abs/2506.04322.

[41] Han Zou, Yuxun Zhou, Jianfei Yang, and Costas J. Spanos. "Device-free occupancy detection and crowd counting in smart buildings with WiFi-enabled IoT". In: *Energy and Buildings* 174 (2018), pp. 309–322. ISSN: 0378-7788. DOI: https://doi.org/10.1016/j.enbuild.2018.06.040.

Appendix A
GitHub Repository

The implementation can be found on Github: https://github.com/brunobastosrodrigues/csi-sense/tree/localization-eval

Appendix B
Figures

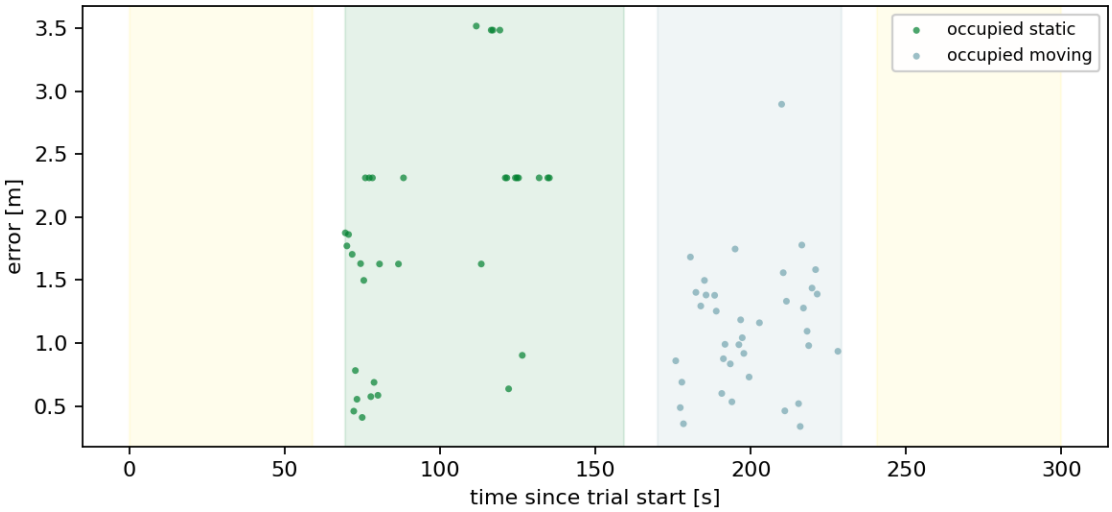

Figure B.1: Single-trial error timeline for T0068 (unfrozen baseline, static pose S1). Phase shading is yellow for empty, green for stationary, blue for moving. The empty phases produce no estimates. Interesting in the static phase is that it produces estimations around 40–70 seconds. This was likely caused by the subject adjusting its posture, which could have re-trigger the localizer for a few samples.

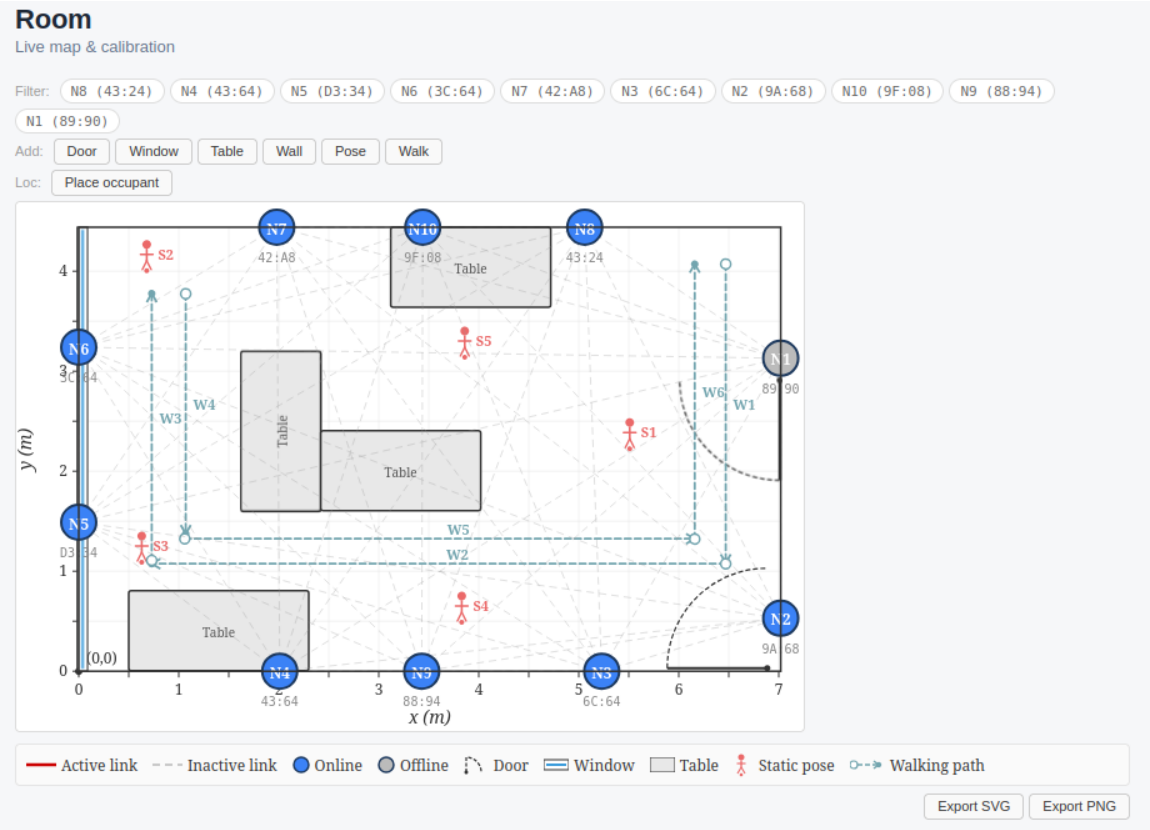

Figure B.2: Monitoring Interface in the application (with one node offline)

Table B.1: Pose Positions

Pose	x	y
S1	5.51	2.28
S2	0.68	4.05
S3	0.63	1.14
S4	3.83	0.54
S5	3.86	3.19

Table B.2: Node Positions

Node	x	y
3C:64	0.00	3.24
42:A8	1.98	4.44
43:24	5.06	4.44
43:64	2.01	0.00
6C:64	5.23	0.00
88:94	3.43	0.00
89:90	7.02	3.13
9A:68	7.02	0.53
9F:08	3.44	4.44
D3:34	0.00	1.49

Declaration of Authorship

‘I hereby declare,

- that I have written this thesis independently;
- that I have written the articles and contributions using only the aids listed in the index;
- that all parts of the thesis produced with the help of aids (incl. AI-Tools) have been precisely declared;
- that I have mentioned all sources used and cited them correctly according to established academic citation rules; this also includes the comprehensible disclosure of all personal publications;
- that I have acquired all immaterial rights to any materials I may have used, such as images or graphics, or that these materials were originally created by myself;
- that I am aware of the legal provisions regarding the publication and dissemination of parts or the entire thesis and that I comply with them accordingly;
- that I am aware that the University will prosecute a violation of this Declaration of Authorship and that disciplinary as well as criminal consequences may result, which may lead to expulsion from the University or to the withdrawal of my title.’

By submitting this thesis, I confirm through my conclusive action that I am submitting the statutory declaration, that I have read and understood it, and that it is true.

St. Gallen, May 19, 2026 Signature:

Declaration of Auxiliary Aids

The creation of this dissertation involved a lot of third-party software, including AI-based tools. I declare all uses of third-party software in the dissertation text that contributed non-trivially to the dissertation's content and are non-standard in such research. Further, by the University of St. Gallen's decree II.B.1.20 section 5.3, I explicitly declare the following uses:

The following tools were used in the completion of this thesis:

Aid	**Usage**	**Affected Parts**
Claude Code	Coding assistance	Code
Claude AI Perplexity Writefull	Spell and grammar checking, rewordings, and paraphrasing	Complete paper
Perplexity AI Claude AI	Finding academic sources	Research work
Claude AI	Creating LaTeX figures	Figures