PHASOR: Enabling New Wireless Sensing Applications with Wi-Fi Phase Synchronization

William Hunter[1,*], Khoa Nguyen[2,*], Kazuhiro Kizaki[3], Shunsuke Saruwatari[3], Dinesh Bharadia[1]
[1]University of California, San Diego [2]VinUniversity [3]Osaka University
{wshunter, dineshb}@ucsd.edu, 23khoa.nm@vinuni.edu.vn, k.kizaki@kg8.so-net.ne.jp, saru@ist.osaka-u.ac.jp

***Abstract*—Wireless infrastructure is now ubiquitously deployed indoors, and reusing it for sensing promises fine-grained physical awareness with no new spectrum or dedicated hardware. However, today's indoor Wi-Fi deployments use access points which are unsynchronized, meaning received channel measurements cannot be effectively combined across multiple receivers, limiting sensing performance. Realizing a full distributed aperture to provide the maximum sensing performance demands tight time and phase synchronization, which until now has required a dedicated wired timing backbone connecting all infrastructure. We present PHASOR (PHase-locked Aperture for Synchronization Over Radio), a system that phase-synchronizes spatially separated commodity Wi-Fi receivers entirely over the air. PHASOR pairs an ESP32 with a controllable TCXO and uses measurements between nodes to gradually align each receiver chain's phase, effectively building a distributed phased array using commodity Wi-Fi chips. We demonstrate that PHASOR can achieve sub-nanosecond time synchronization over a standard Wi-Fi link, and that this synchronization can be used to build an accurate indoor location system. PHASOR also provides a high-precision synchronized clock source for external devices like network switches and SDRs. We open-source the full hardware and software design of PHASOR for the research community.**

***Index Terms*—Wi-Fi sensing, integrated sensing and communication (ISAC), phase synchronization, distributed antenna systems, time-difference-of-arrival (TDoA), CSI**

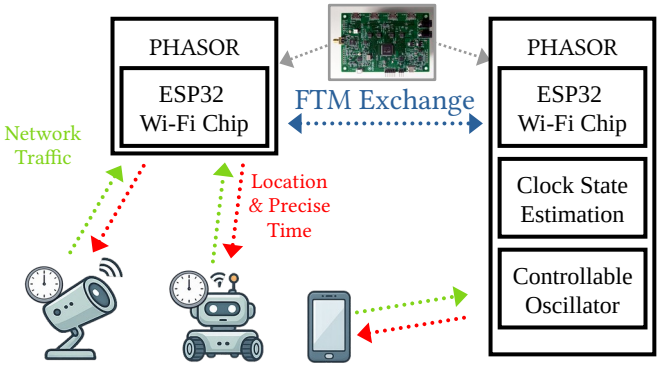


Fig. 1: Overview of PHASOR. Independent clocks cause spatially distributed commodity Wi-Fi receivers to make noncoherent measurements of transmitted packets, making them infeasible for localization. PHASOR nodes periodically exchange standard FTM packets and phase-align their clocks. Once synchronized, the receivers can provide a common time and location source to everything on the network.

I. Introduction

Many physical and digital tasks are no longer performed by one device. Robots, sensors, edge computers, and personal devices increasingly observe and act in the same environment, exchanging measurements and commands over a network. Collaborative robotics and Wi-Fi-assisted mapping already illustrate how observations from multiple devices can be combined [1], [2]. However, carrying data between devices does not place their observations and actions in a common physical frame. That physical frame has both a temporal component and a spatial component. To determine whether observations from different devices describe the same event, the devices must know *when* the observations were made. To interact with, navigate relative to, or reason about a transmitting device, they may also need to know *where* that device is. These requirements arise naturally when distributed networked devices interact in the physical world: their measurements and actions must be related in time, and often in space, before they can be combined.

Thus, a common space- and time-reference frame is critical for networked systems. Tight time synchronization can improve consensus and scheduling in a network, but can also enable spatial awareness. When multiple receivers share a physical timebase, they can make coherent channel measurements, allowing the receivers to constrain the transmitter's location [3]–[6]. Thus, a shared timebase between nodes in a wireless network enables physical awareness. Since 1 nanosecond of timing error corresponds to $\sim$ 30cm of ranging error, around the state-of-the-art for indoor Wi-Fi positioning accuracy, we identify sub-nanosecond timing errors as a suitable requirement for precision wireless time synchronization.

In this work, we aim to build a precise time synchronization system over Wi-Fi. To date, commodity Wi-Fi synchronization works have focused primarily on software-visible clocks. PTP-style systems and Wi-Fi synchronization protocols can estimate and correct clock offset [7]–[9], but they do not necessarily discipline the physical reference that drives packet timestamps, sampling, and carrier generation. Systems that do provide sub-nanosecond synchronization generally add a separate synchronization facility. RFClock uses a specialized dual-tone frequency signal and a separate UWB timing path [10]; Pulsar combines UWB propagation measurements with specialized stable clocks [11]; and AirSync uses dedicated over-the-air reference transmissions with FPGA-class SDR front-ends

*These authors contributed equally to this work.

[12]. Some existing works have focused only on coherent location sensing by using dedicated wired time-distribution architecture [13]–[16]. To date, precise physical time at the accuracy required for coherent sensing has required specialized infrastructure which is difficult to deploy at network scale.

We present PHASOR (PHase-locked Aperture for Synchronization Over Radio), a self-contained Wi-Fi synchronization system that uses standard FTM exchanges and CSI as feedback for disciplining each receiver's common clock reference. PHASOR uses an ESP32 receiver capable of measuring channel-state information and hardware reception timestamps, clocked by a controllable oscillator allowing fine-grained control of the ESP32's LO. Multiple PHASOR nodes repeatedly exchange packets in order to estimate their mutual time and LO phase offsets, and adjust their oscillator frequency to bring both into alignment. PHASOR uses this synchronization to make phase-coherent measurements of Wi-Fi packets to provide a precise location, and additionally provides output timing reference signals for use with other devices, such as network switches or 5G RUs. The same Wi-Fi fabric can therefore support synchronized operation and passive spatial awareness without GPS, a wired timing backbone, or a specialized over-the-air waveform. As an application-level validation, PHASOR implements a Wi-Fi sensing scheme that demands tight time synchronization, using time-difference-of-arrival to localize unmodified transmitters from ordinary Wi-Fi packets with median errors of 2.48–3.95 m across three environments.

The main contributions of this paper are as follows:

- We introduce a self-contained Wi-Fi synchronization architecture that phase-aligns spatially separated commodity receivers using only standards-compliant FTM exchanges, delivering both a shared timebase and device localization from the same infrastructure.
- We build a commodity Wi-Fi node in which a single reference oscillator driving packet timestamps, sampling, and RF carrier generation, is physically steerable in software, and which also exports standard 10MHz and 1PPS signals to discipline external devices such as network switches and SDRs.
- We drive this hardware with a closed-loop controller combining FTM-based Kalman acquisition of time and fractional-frequency offset that is robust to dropped and delayed exchanges, CSI-assisted fine phase tracking for receiver LO alignment, and latency- and constraint-aware physical clock control, together with controller-state correction of residual time-varying phase in cross-receiver CSI.
- We implement and evaluate the complete PHASOR prototype, demonstrating sub-nanosecond inter-node clock alignment down to 15dB SNR, characterizing relative LO-phase stability under missing observations and synchronization outages, and validating the synchronized hardware timestamps through passive reverse-TDoA localization of unmodified transmitters from ordinary Wi-Fi packets.

II. Related Work

A. Wireless Time Synchronization

Several existing works have explored wireless time and frequency synchronization. These systems generally rely on ultra-wideband (UWB) due to its high time-bandwidth product, or custom waveforms. RFClock [10] distributes a dual-tone carrier for frequency and a UWB pulse for a pulse-per-second reference. Pulsar [11], [12] pairs UWB ranging with a chip-scale atomic clock for propagation-aware synchronization A larger family of UWB systems [3], [17], [18] treat synchronization as a byproduct of localization; these achieve fine time alignment but do not discipline the node's own frequency reference. All of these systems are additionally limited to low ranges due to their employment of UWB. Some effort has been made in over-the-air synchronization using commodity Wi-Fi [7], [8], but since these systems use MAC-layer timestamping, they do not provide the tight nanosecond-level synchronization required for coherent reception. Since PHASOR disciplines its oscillator at the hardware level, it can provide nanosecond-accurate synchronization using standard Wi-Fi packet.

B. Channel Measurement Tools

Several non-synchronized systems have been designed to make Wi-Fi channel measurements for sensing. Similar to PHASOR, these works all exploit Wi-Fi chipsets which expose Channel-State Information (CSI) to the end user. ESPARGOS [19] builds a single array of 8 ESP32s with a shared clock to make a multi-antenna channel measurement. Other works [20], [21] provide single-antenna CSI measurements using the ESP32. Multi-antenna CSI can also be obtained from other platforms including Intel [22], [23] and Broadcom [24] chipsets. These systems can be used for non-coherent sensing only, since they do not provide fine-grained synchronization between receivers.

There are existing works that provide phase coherence across receivers, at a higher price point. DICHASUS [13] builds a large phased array using PlutoSDR receivers and synchronizes several arrays over-the-air using a dedicated synchronization signal. Sandra et al. [14] use USRP X410 receivers with GPS-based phase synchronization to provide a coherent distributed receiver testbed. Other testbeds [15], [16] use dedicated cabling for synchronization between receivers. AirSync [12] uses an FPGA-based Wi-Fi receiver to perform instantaneous carrier-phase corrections, enabling coherent Wi-Fi transmission. These works provide similar channel measurement capabilities to PHASOR, but at significantly increased cost since they use dedicated software-defined-radio (SDR) receivers. Additionally, they require either a dedicated hardware backbone for synchronization, further increasing deployment cost and complexity, or a customized air-interface protocol. PHASOR obtains time and phase synchronization using standard Wi-Fi packets and commodity hardware.

C. TDoA Localization

We demonstrate PHASOR's practicality for time-sensitive tasks by benchmarking TDoA localization, an algorithm which requires nanosecond-accurate timestamping.

Several works have implemented TDoA localization over Wi-Fi [4], [5], however these works implement the receiver on SDR-based hardware using an FPGA for signal processing, and synchronize the receivers by listening to packets broadcast from known, static reference devices. While these systems establish that Wi-Fi TDoA is feasible, they rely on wired synchronization backbones to cancel receiver offset.

Several works attempt to estimate receiver clock offset and device position simultaneously. Golden et al. [6] combine both of Wi-Fi's timing modalities on an FPGA receiver, implementing FTM's two-way ranging alongside passive TDoA and using the FTM exchanges to calibrate the inter-receiver time offset for TDoA-only receivers. [25], [26] assume that clock offsets are relatively stable over short periods of time, and perform explicit joint estimation of receiver clock offsets and target position over time. These works either require explicit cooperation at the UE, or make assumptions that UEs transmit continuously for long periods of time and dynamically move through the environment.

A recent parallel line of work performs passive localization directly on the user device, now standardized as the passive ranging mode of IEEE 802.11az [27]. Here the device estimates its own position by overhearing synchronized transmissions exchanged between fixed base stations, whether through cooperative FTM among anchors [28] or by passively listening to the round-trip FTM exchange [29]. Because these schemes recover only per-link range estimates rather than an entire phase-synchronized channel, they are likely to achieve lower accuracy; moreover, they depend on every access point meeting tight timing requirements exactly, which requires tight coordination between access points at the PHY layer. PHASOR instead disciplines each receiver's oscillator to deliver a coherent, phase-synchronized channel measurement of arbitrary transmitters.

III. System Model

A. Network Model

PHASOR's goal is to establish a network of time- and phase-synchronized Wi-Fi receivers. Each PHASOR node is a Wi-Fi transceiver distributed throughout the environment, and PHASOR nodes synchronize to each other by making periodic two-way FTM (802.11mc [30]) packet exchanges to accurately measure their clock and LO phase offset. We adopt a leader-follower architecture in which one node leaves its clock free-running, and all other nodes exchange with it periodically. PHASOR nodes localize devices on the network by sniffing for transmitted Wi-Fi packets and measuring the reception time and Channel State Information (CSI). We provide a brief model for time offset measurement with FTM and the wireless channel below. Throughout, i indexes a receiver, j a transmitter, k a subcarrier, and n a packet.

B. Two-Way Timestamp Exchange

PHASOR uses a two-way packet exchange between nodes to measure timing offsets. Node j transmits at local time t_1; node i receives at local t_2, replies at local t_3, and j receives at local t_4. Under reciprocal propagation delay τ_{p},

$$\hat{\theta}_{ij} = \frac{(t_2 - t_1) - (t_4 - t_3)}{2}, \qquad \widehat{\mathrm{RTT}} = (t_4 - t_1) - (t_3 - t_2). \tag{1}$$

This exchange is implemented by Wi-Fi's FTM standard, in which t_1–t_4 are captured in hardware at the PHY layer, giving them very high accuracy. However, measuring the time offset with high accuracy is not enough, as each device's oscillator will continuously drift due to thermal noise. Thus, we must continuously estimate both the time offset and its drift to fully phase-align two receivers. PHASOR additionally notes that the CSI recorded from FTM exchanges themselves provides further information about the time offsets.

C. Channel Measurement Model

Receiver i estimates the channel from transmitter j on each received packet n. Writing the complex baseband channel estimate as a function of baseband frequency f from the carrier f_c,

$$H_{ij}(f,n) = \underbrace{e^{\jmath(\phi_j - \phi_i)}}_{\text{Const. bias}} \underbrace{e^{\jmath 2\pi \Delta f_{ij}\, t_n}}_{\text{CFO}} \underbrace{e^{-\jmath 2\pi (f_c+f)\, \delta_{ij}(n)}}_{\text{clock/timing offset}} \underbrace{\times \sum_p \alpha_p\, e^{-\jmath 2\pi (f_c+f)\tau_p}}_{\text{Real Channel}} \tag{2}$$

Where α_p are per-path complex gains. We can see 3 error terms which corrupt our ability to measure the true channel: A constant bias term $e^{\jmath(\phi_j - \phi_i)}$ caused by the receiver and transmitter's initial PLL lock phase, a packet-dependent phase bias $e^{\jmath 2\pi \Delta f_{ij}\, t_n}$ caused by residual CFO Δf_{ij}, and a sampling time offset caused by clock mismatch between i and j, $\delta_{ij}(n) = \theta_i(t_n) - \theta_j(t_n)$. Note that, on the ESP32 $H_{ij}(f,n)$ is measured with respect to the reception timestamp measured in hardware. PHASOR must remove all of these factors in order to make phase-coherent receptions across multiple receivers.

D. Clock Model and Derived References

We model each node's local clock by its time offset $\delta_i(t)$ and fractional frequency offset $\varepsilon_i(t)$ relative to the leader. Over an interval Δ,

$$\begin{bmatrix} \delta_i(t+\Delta) \\ \varepsilon_i(t+\Delta) \end{bmatrix} = \begin{bmatrix} 1 & \Delta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \delta_i(t) \\ \varepsilon_i(t) \end{bmatrix} + \begin{bmatrix} w_\delta \\ w_\varepsilon \end{bmatrix}, \tag{3}$$

where w_δ and w_ε capture white and random-walk frequency noise.

Critically, because all clocks used in PHASOR are derived from the same reference oscillator, both the radio's LO and sampling clock inherit the same fractional error ε_i, scaled by fixed ratios:

$$\Delta f_i = f_c\, \varepsilon_i, \qquad \Delta f_i^{\mathrm{s}} = f_{\mathrm{s}}\, \varepsilon_i. \tag{4}$$

where f_c is the carrier frequency and f_s is the sampling frequency, and Δf_i and Δf_i^s are the carrier frequency offset (CFO) and sampling frequency offset (SFO), respectively. Thus, PHASOR can change only the reference oscillator's frequency to discipline both the radio's CFO and SFO simultaneously, and furthermore, we can use the LO phase between two PHASOR nodes to provide a fine-grained measurement of their time offset. We exploit this fact in §IV-D to estimate δ_i at a finer resolution than can be provided by hardware timestamps.

IV. System Design

A. Overview

PHASOR's goal is to synchronize multiple radio receivers throughout the environment to within 1ns. PHASOR accomplishes this synchronization using two-way FTM exchanges coupled with signal processing and control techniques discussed below. Once the nodes are synchronized, each node can listen and receive sensing packets, reports the resulting channel estimates to a backend server over a separate wireless link. The nodes stop listening and send a new set of FTM exchanges every T_{sync} seconds to measure the clock offset and iteratively cancel clock drift, maintaining synchronization. For deployment simplicity, in this work one node is designated as the leader, and all other nodes attempt to synchronize to it. Figure 2 demonstrates the flow of data through the system and PHASOR's clock control loop.

Concretely, each PHASOR node is built on a software-steerable physical clock (§V-B) and closes a control loop around it that acquires a coarse time-and-frequency offset from FTM hardware timestamps via a drop-tolerant Kalman filter (§IV-B), actuates the oscillator with a latency- and excursion-aware model-predictive controller (§IV-C), sharpens this estimate below the hardware-timestamp floor using CSI carrier phase (§IV-D), and calibrates the residual lock-phase bias to deliver phase-coherent measurements across nodes (§IV-E).

B. Clock Offset Estimation

PHASOR performs FTM exchanges to synchronize each node, but these exchanges can occasionally be delayed or fail, and are subject to measurement noise. We therefore explicitly model the clock's state using a Kalman filter in order to optimally control it.

Follower node i maintains the state

$$\mathbf{x}_i = \begin{bmatrix}\delta_i & \varepsilon_i\end{bmatrix}^\top, \tag{5}$$

the time and fractional frequency offset relative to leader 0, which propagates according to the clock model of (3). The elapsed interval between measurements can vary due to dropped packets or networking delays, which, without being handled, would cause model mismatch and suboptimal control. Thus, we define the inter-measurement time $\Delta_n = t_n - t_{n-1}$ and explicitly include it in the system model and noise covariance matrices. The state update equation for the Kalman filter at node i is then:

$$\begin{aligned}
\mathbf{x}_i(t_n) &= \mathbf{A}(\Delta_n)\,\mathbf{x}_i(t_{n-1}) + \mathbf{B}(\Delta_n, \tau_n^{\text{lag}})\Delta u_i(n) + \mathbf{w}_n \\
\mathbf{w}_n &\sim \mathcal{N}\big(\mathbf{0},\, \mathbf{Q}(\Delta_n)\big), \\
\mathbf{A}(\Delta_n) &= \begin{bmatrix}1 & \Delta_n \\ 0 & 1\end{bmatrix}, \\
\mathbf{Q}(\Delta_n) &= \begin{bmatrix}q_\delta \Delta_n + q_\varepsilon \Delta_n^3/3 & q_\varepsilon \Delta_n^2/2 \\ q_\varepsilon \Delta_n^2/2 & q_\varepsilon \Delta_n\end{bmatrix}
\end{aligned} \tag{6}$$

Where $\mathbf{Q}(\Delta_n)$ is the process noise covariance, for which we adopt the standard clock-servo noise model from [31] with q_δ, q_ε chosen based on oscillator quality. $\mathbf{B}(\Delta_n, \tau_n^{\text{lag}})$ and $\Delta u_i(n)$ are control inputs described below. Each FTM exchange supplies an observation of the state:

$$\begin{aligned}
\hat{\theta}_n &= \mathbf{C}\,\mathbf{x}_i(t_n) + v_n, \qquad v_n \sim \mathcal{N}(0, \sigma_\theta^2), \\
\mathbf{C} &= \begin{bmatrix}1 & 0\end{bmatrix}.
\end{aligned} \tag{7}$$

with σ_θ^2 being the FTM measurement variance, chosen empirically.

The Kalman filter updates its estimate $\mathbf{x}_i(t_n)$, and uses this estimate to update each node's internal clock bias, described below in §IV-C. However, due to processing latency and latency in the communication between the CPU and the clock, this update is not applied immediately at the measurement time, instead only taking effect at time $t_n + \tau_n^{\text{lag}}$ and this must be accounted for in the state model. Let $u_i(t_n)$ denote this applied tuning word, applied τ_n^{lag} seconds after t_n. Since $\hat{\varepsilon}_{i0}(t_{n-1})$ already includes the effect of $u_i(t_{n-1})$, only the difference in control inputs $\Delta u_i(n)$ enters the state model:

$$\mathbf{B}(\Delta_n, \tau_n) = \begin{bmatrix}\Delta_n - \tau_n^{\text{lag}} \\ 1\end{bmatrix} \tag{8}$$

A standard linear Kalman update with (6) gives the posterior. If an FTM measurement fails, n is simply not updated, so t_n only measures the times of successful measurements. Since the process and noise models explicitly include the time difference between measurements, the Kalman filter's model remains consistent.

C. Disciplining the Oscillator

Given the Kalman filter's estimate of $\hat{\mathbf{x}}_i$, the node must drive both components to zero. PHASOR performs this by using a controllable oscillator, for which we can add a fractional bias $u_i(t_n)$:

$$\varepsilon_i(u_i) = \varepsilon_i^0 + u_i, \qquad |u_i| \le u_{\max}, \tag{9}$$

Our goal is to choose a controller which programs u_i to synchronize each node's clock. There are two key constraints for this controller: First, the controller must be capable of handling non-uniform measurement intervals. Second, and most importantly, changing ε to cancel a time offset injects a phase ramp into the received channel measurements; the cost function must therefore trade time convergence rate against the frequency excursion used to achieve it.

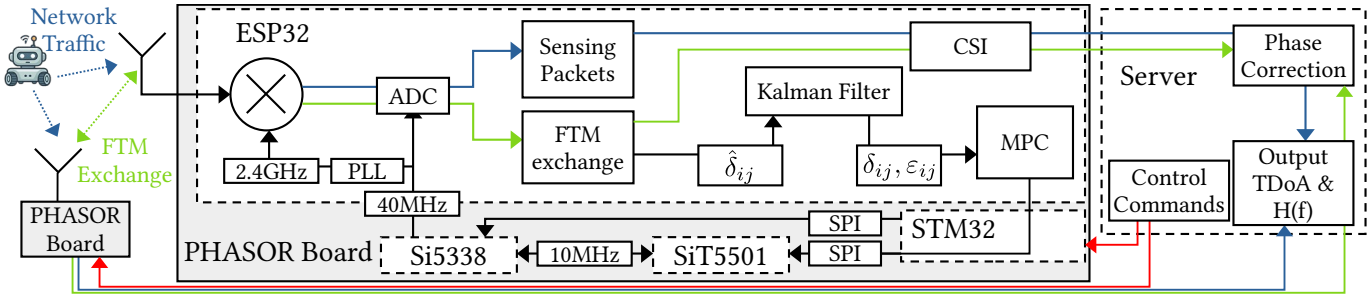


Fig. 2: PHASOR data flow. The central board consists of an ESP32 clocked from a high-precision oscillator. Each node makes periodic FTM exchanges with other PHASOR nodes, measuring an instantaneous time offset. These measurements are fed into a Kalman filter (§IV-B) and used to synchronize each board's oscillator in time and frequency (§IV-C). Channel phase between nodes is used to refine timestamp accuracy (§IV-D) and make phase-coherent measurements of sensing packets (§IV-E)

A standard controller like PI cannot meet these constraints, as it does not consider a varying measurement interval or limit frequency excursions. However, since the state space update equation (6) is relatively simple, we can directly invert it in order to drive $\hat{\mathbf{x}}_i$ to $\mathbf{0}$, a so-called Model-Predictive Controller (MPC), allowing PHASOR to directly account for the measurement interval and frequency excursion.

The posterior estimate is referenced to the FTM epoch t_n, whereas a frequency update command reaches the oscillator only after processing and SPI delay, meaning the oscillator is actually updated at $t_a = t_n + \tau_n^{\text{lag}}$. Let $u_i(t_a)$ be the tuning word already applied at t_a, and let ε_c be a candidate frequency offset we wish to apply. Assuming a constant rate of FTM exchanges T_h, our goal at each step is to choose a control input which drives $\hat{\mathbf{x}}_i$ to 0 at the next exchange. Thus, the predicted time offset at $t_a + T_h$ is $\hat{\delta}_i(t_a) + T_h\varepsilon_c$. Motivated by the required balance between time and frequency offset mentioned above, we choose the following criterion to decide the optimal control input:

$$\varepsilon_c^\star = \arg\min_{\varepsilon_c} \underbrace{\left[\hat{\delta}_i(t_a) + T_h\varepsilon_c\right]^2}_{\text{Residual time offset}} + \underbrace{\lambda_f \left(T_h\varepsilon_c\right)^2}_{\text{Frequency excursion}} + \underbrace{\lambda_\Delta T_h^2 \left[\varepsilon_c - \hat{\varepsilon}_i(t_a)\right]^2}_{\text{Frequency change}}. \tag{10}$$

Here $\lambda_f \geq 0$ weights the frequency excursion and $\lambda_\Delta \geq 0$ weights the change from the currently estimated frequency offset. Since (10) is scalar and quadratic, its unconstrained minimizer is

$$\varepsilon_c^\star = \frac{-\hat{\delta}_i(t_a) + \lambda_\Delta T_h \hat{\varepsilon}_i(t_a)}{T_h\left(1 + \lambda_f + \lambda_\Delta\right)}. \tag{11}$$

Using (9), the corresponding nominal tuning word is

$$\widetilde{u}_i = u_i(t_a) + \varepsilon_c^\star - \hat{\varepsilon}_i(t_a). \tag{12}$$

Before being issued, $\widetilde{u}_i$ is clipped to the TCXO range, slew-limited, and quantized.

D. Fine-grained Phase Control

The FTM loop of § IV-B resolves δ_{ij} only to the hardware timestamp quantization ($\sim$0.5 ns), yet a far finer observation of the same quantity is already present in the channel measurement. Isolating the clock term of (3) at the carrier, the received phase rotates as $e^{-\jmath 2\pi f_c\, \delta_{ij}(n)}$, so the clock offset appears scaled by the full carrier frequency. One carrier cycle corresponds to $T_c = 1/f_c \approx 406$ ps at 2.462 GHz, giving an observation of δ_{ij} whose resolution lies well below the FTM floor. The price is ambiguity: phase is known only modulo T_c, and the constant lock-phase bias $e^{\jmath(\phi_j - \phi_i)}$ together with the propagation and RF-chain delays in (3) are indistinguishable from clock offset on a single link. The carrier-phase loop therefore observes changes in δ_{ij} at high resolution rather than its absolute value.

Within one FTM burst, several packets arrive from the same peer over a static channel. At each follower i, we take the first received CSI from the leader $H_{0i}(f, n_0)$ as a reference and, for each later packet n, form the per-subcarrier conjugate product, unit-normalize, and sum coherently over the used subcarriers $k \in \mathcal{K}$:

$$\Psi_n \;=\; \sum_{k\in\mathcal{K}} \frac{H_{0i}(f_k, n)\, H_{0i}^*(f_k, n_0)}{\left|H_{0i}(f_k, n)\, H_{0i}^*(f_k, n_0)\right|}. \tag{13}$$

The static channel $\sum_p \alpha_p e^{-\jmath 2\pi (f_c + f)\tau_p}$ and the constant bias $e^{\jmath(\phi_i - \phi_0)}$ of (3) are common to both packets and cancel in the product, leaving only the clock-driven rotation:

$$\psi_n \;=\; \arg\Psi_n \;\approx\; -2\pi f_c\big[\delta_i(n) - \delta_i(n_0)\big] \tag{14}$$

Converting to time, $z_n^\phi = -\psi_n/(2\pi f_c)$ is an observation of $\delta_i(n) - \delta_i(0)$, sharing the same measurement matrix as for time offset $\mathbf{C} = [1 \;\; 0]$. The cycle ambiguity is resolved by unwrapping z_n^ϕ against the Kalman prediction $\hat{\delta}_i$ and rounding to the nearest multiple of T_c. In order to avoid an unwrapping error, the rounded value is trusted only when 3σ of the predicted offset plus a small guard is below $T_c/2$, otherwise the loop re-anchors rather than risk a cycle slip. When valid, z_n^ϕ enters the Kalman update from equation 6 with variance σ_ϕ^2 derived from R_n and the phase spread, tightening $\hat{\mathbf{x}}_i$ beyond the FTM floor. Section VI-C1 evaluates the additional phase observation's effect on timing accuracy.

E. Received Phase Calibration

Although PHASOR provides a common time reference across all receivers, the per-receiver lock-phase bias $e^{j\phi_i}$ of (2) remains: each node's PLL acquires an arbitrary initial phase, and this bias is indistinguishable from propagation phase on any single link. Coherent combination of channel measurements across nodes requires knowing the relative bias $(\phi_i - \phi_j)$ between each pair of receivers.

We estimate it in-band using a reciprocal exchange between the two PHASOR nodes themselves. Node i transmits to j and j replies to i within the same burst, and we form the conjugate product of the two directions' channel estimates at a common subcarrier. Applying (2),

$$\begin{aligned}&\arg\big(H_{ij}(f,n_1)\,H^*_{ji}(f,n_2)\big) \approx \\ &2(\phi_j-\phi_i) - 2\pi f_c\big[\delta_{ij}(n_1)+\delta_{ij}(n_2)\big].\end{aligned} \tag{15}$$

By reciprocity, the propagation terms form $|C(f)|^2$, which is real and positive at every subcarrier, so the channel drops out entirely. The lock phases, in contrast, add rather than cancel, leaving twice the bias. The remaining clock term is small and is known from the synchronization controller's own state (§IV-D), so it can be subtracted directly. Dividing by two then recovers $(\phi_i - \phi_j)$ up to a half-cycle ambiguity, which must be resolved by measuring a packet from a known location.

Because the calibration exchange runs between PHASOR nodes over the air and reuses controller state, relative receiver phase is recovered with no wired reference and no cooperation from external transmitters. The estimate remains valid until either PLL re-locks, at which point the exchange is simply repeated. §VI-C3 evaluates the stability of the calibrated inter-node phase for a static transmitter.

V. Hardware Design

A. Prototype Architecture and Clock Domains

A PHASOR node integrates an ESP32-S2 Wi-Fi receiver, an SiT5501 digitally controlled temperature-compensated crystal oscillator (DCTCXO), an Si5338L clock generator, and an STM32F746 control MCU. Figure 3 shows the corresponding processing, clock, and control paths. The ESP32-S2 performs all runtime synchronization processing and CSI acquisition. The STM32F746 initializes the ESP32-S2 at boot and, during runtime, serves as a low-level interface between the ESP32-S2 and the SiT5501.

The SiT5501 generates a nominal 10-MHz reference and is configured with a pull range of ± 6.25 ppm. Under this configuration, it accepts a signed 26-bit two's-complement frequency-control word with a nominal fractional-frequency resolution of 5×10^{-12}. Runtime frequency corrections are applied to the SiT5501, and the Si5338 synthesizes the 40-MHz external reference required by the ESP32-S2 from the adjusted 10-MHz signal. Because the ESP32-S2 derives both its RF synthesizer and sampling subsystem from the 40-MHz reference, steering the SiT5501 jointly affects the CFO and SFO modeled in Section III-D.

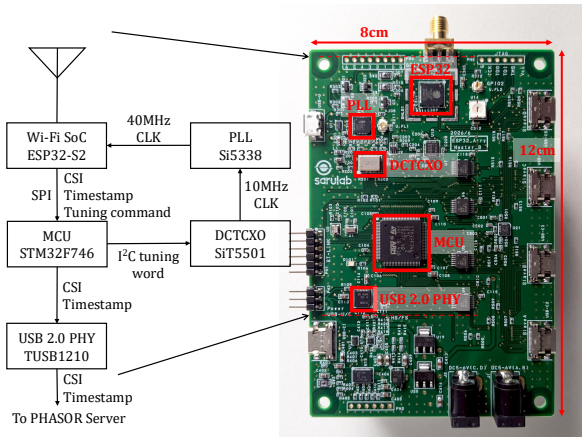


Fig. 3: PHASOR node hardware, clock, and control architecture. The ESP32-S2 executes the FTM-based synchronization estimator and controller and sends oscillator commands to the STM32F746. The STM32F746 applies these commands to the SiT5501 over I^{2}C, and the Si5338 synthesizes the 40-MHz ESP32-S2 reference from the frequency-adjusted 10-MHz output. Spatially separated nodes synchronize through over-the-air IEEE 802.11mc FTM exchanges and share no wired timing reference.

The component-only bill of materials for one prototype board is approximately \$151, including \$71 for the SiT5501, \$24 for the Si5338, \$17 for the STM32F746, and \$3 for the ESP32-S2. The cost of this BoM is dominated by the controllable oscillator, a relatively rarely-used part, which could become significantly cheaper as time-synchronized networking becomes more prevalent. This estimate excludes PCB fabrication, assembly, optional host-side collection hardware, external power supplies, and cables.

B. Closed-Loop Clock-Control Realization

All synchronization processing runs on the ESP32-S2, including FTM handling, mode-specific bias correction, Kalman filtering, MPC, and oscillator-command generation. The STM32F746 performs no estimation or control optimization; it receives each command from the ESP32-S2 and writes the corresponding 26-bit control word to the SiT5501 over I^{2}C. It is required because the TCXO and PLL must be programmed at boot to generate the ESP32's 40MHz clock; the ESP32 cannot bootstrap itself.

During runtime, the STM32F746 relays the ESP32-S2's oscillator commands to the SiT5501. For each update, the STM32F746 writes the corresponding 26-bit frequency-control value to the SiT5501 over I^{2}C. The lower 16-bit word is written first, followed by the upper 10-bit word; loading the upper word initiates the frequency change. The resulting adjustment propagates through the Si5338 to the 40-MHz reference supplied to the ESP32-S2, closing the loop from an FTM observation to a physical change in the receiver's common clock source.

In order to accurately model the clock state, it is necessary to estimate the time at which the STM32 actually applies the frequency-change command to the controller in the ESP's reference frame. This is accomplished by having the ESP32 measure the total round-trip time between sending a SPI command and receiving a 'write complete' message from the STM32, with the STM32 measuring its own receive and transmit times as well as the time it wrote the command to the SiT5501. This permits estimation of the write time in the ESP's reference frame, τ_n^{lag} in equation 8.

C. Synchronization and Sensing Pipeline

PHASOR uses the leader–follower topology defined in Section IV, with one FTM session per follower in each synchronization round. Between synchronization updates, each ESP32-S2 acquires CSI together with the associated hardware reception timestamps and packet metadata. Each PHASOR node exports these records over USB to a host-side collection endpoint. In a compact deployment, the node may be connected directly to the central server. For spatially distributed deployments, an optional Raspberry Pi may instead receive the USB stream and relay it over Ethernet to the central server. The choice of data-transport path does not affect synchronization: all FTM processing, clock-state estimation, MPC, and oscillator control remain local to the ESP32-S2 and STM32F746, and neither USB nor Ethernet carries a timing reference.

The central server associates reports from different ESP32-S2 receivers with the same over-the-air packet, computes channel impulse responses from the received CSI, and estimates the inter-node channel phase and residual phase bias. For PDoA processing, the server compares CSI across the relevant links and applies the phase-calibration method described in Section IV-E. The server then constructs the cross-node channel observations used by the TDoA and PDoA applications in Section VI. Section VI-C2 evaluates the airtime and sensing interruption introduced by periodic FTM exchanges, together with the tradeoff between synchronization frequency and residual clock drift. A synchronization interval is performed every 10 seconds by default, in which all follower nodes perform one FTM exchange to the leader node.

D. Clock Outputs

PHASOR also provides a synchronized 10MHz frequency reference and 1PPS clock output, the standard timing reference signals, to provide to other devices and for verification purposes. The 10MHz signal is generated directly from the Si5338 PLL and output via a U.FL port, as its purpose is only to be frequency-accurate and is not phase-aligned. Generation of the 1Hz 1PPS signal is slightly more complex, as it must be phase-locked between receivers. PHASOR only synchronizes the ESP32's internal MAC clock, and generating a 1PPS edge from it is nontrivial: the MAC clock is readable only through a software register and cannot drive any pin or peripheral directly. The CPU cycle counter (CCOUNT), however, derives from the same disciplined reference at a fixed ratio (160 cycles per MAC microsecond at the 160-MHz CPU clock), so it serves as a hardware bridge. At initialization, PHASOR calibrates the MAC-to-CCOUNT mapping by bracketing MAC microsecond transitions with CCOUNT reads and taking the median transition midpoint over 256 samples. A timed interrupt is then used to generate the 1PPS output signal based on the CCOUNT value. This method allows PHASOR to produce a high-accuracy PPS output from software without using an FPGA.

VI. Evaluation

We test PHASOR across a variety of indoor and outdoor scenarios demonstrated in figure 5. We perform 3-node tests of PHASOR's sensing capability by moving a transmitter throughout the space and predicting its position, and we verify PHASOR's synchronization accuracy by making wired measurements of its clock outputs using long cables.

A. Core Time-Sync Performance

We verify PHASOR's performance by measuring the variation of each board's 10MHz and 1PPS phase offset over time on a common timebase. We accomplish this by connecting each node's 10MHz/1PPS outputs to a Calnex Sentinel [32] which measures the relative phase offset of each signal. We quantify synchronization stability using the overlapping Allan deviation $\sigma_y(\tau)$, the standard metric for fractional-frequency stability [31]. It is computed from the fractional frequency $y = \Delta f / f_0$ averaged over successive intervals of length τ as

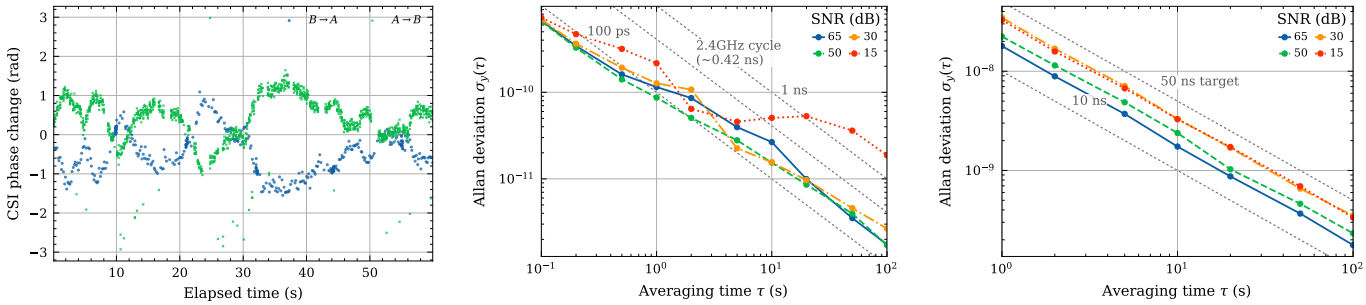
$$\sigma_y(\tau) = \sqrt{\tfrac{1}{2}\left\langle (\bar{y}_{n+1} - \bar{y}_n)^2 \right\rangle}, \tag{16}$$

Because a lower $\sigma_y(\tau)$ at a given τ means two nodes' clocks stay more tightly aligned over that averaging interval, it directly captures the coherence PHASOR must maintain for cross-receiver channel combination. Since PHASOR actively disciplines its oscillators, the phase error cannot walk arbitrarily far from zero, and the Allan deviation decreases as the interval increases.

In figures 4b and 4c, we plot the Allan deviation of the 10MHz and 1PPS outputs with varying SNR levels at the receiver, demonstrating timestamping performance down to 15dB SNR. The 10MHz oscillator has an error less than 1ns over the entire experiment, degrading slightly at low SNR. In figure 4a, we plot the phase difference between two PHASOR nodes' LOs over a 1 minute period, demonstrating that they stay within 2 radians of each other over the entire period. Figure 6 plots a time-series of the 1PPS output over 10 minutes, demonstrating that the 1PPS remains tightly synchronized.

B. End-to-End: TDoA Localization

As described in Section I, once PHASOR's receivers share a common timebase, the arrival times of a single ordinary Wi-Fi packet at different receivers become a direct geometric constraint on the transmitter's location. This is the principle of time-difference-of-arrival (TDoA): because all receivers are synchronized, the difference between the timestamps recorded at two receivers depends only on the difference in their distances



(a) (b) (c)

Fig. 4: Core time-sync performance. (a) Estimated LO phase offset using CSI at the leader and follower, phase does not drift by more than 2 radians over the 1 minute. (b) 10 MHz Allan deviation across SNR levels; (c) 1 PPS Allan deviation across the same sweep. Note that the 1PPS's deviation is higher, around the 10ns mark, due to CPU latency in the ESP. Allan deviation of the 10MHz LO is also reported with a constant phase bias subtracted as it is impossible to

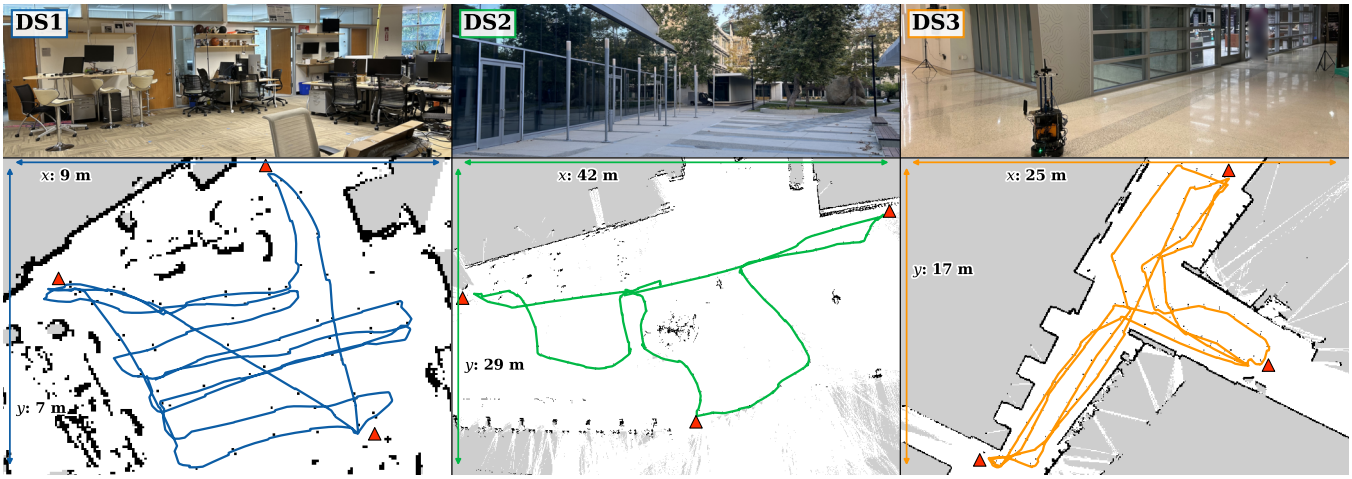


Fig. 5: Environments for 3 TDoA Location datasets. DS1 is an indoor scenario with continuous line-of-sight to all 3 PHASOR nodes, spanning 9x7 meters. DS2 is a long-range outdoor scenario with LoS occasionally obstructed by light-posts and trees. DS3 is an indoor scenario with LoS to at most one node at a time.

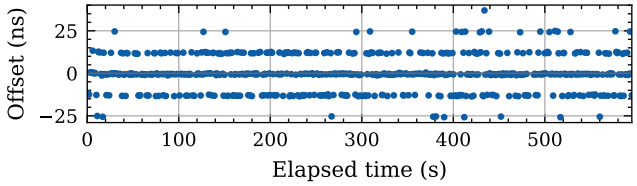


Fig. 6: Time-series error between PPS outputs measured by the sentinel. The PPS converges to within 1-2 nanoseconds, but 12.5ns of noise is added by CPU cycle latency on some pulses.

to the transmitter, not on any unknown transmit time. For a transmitter at unknown position $\mathbf{p}$ observed by receivers i and ℓ at known positions $\mathbf{r}_i$ and $\mathbf{r}_\ell$, the measured timestamp difference satisfies

$$c\,(\hat{t}_i - \hat{t}_\ell) = \|\mathbf{p} - \mathbf{r}_i\| - \|\mathbf{p} - \mathbf{r}_\ell\|, \tag{17}$$

where c is the speed of light and $\hat{t}_i$ is the synchronized reception timestamp at receiver i. Each receiver pair defines a hyperbola (in 2-D) on which the transmitter must lie, and the intersection of constraints from multiple pairs yields the transmitter's position. With 3 synchronized receivers, we can uniquely solve for p using the reception timestamp at all 3 receivers.

We emphasize that any residual synchronization error $\delta_{i\ell}$ between receivers maps directly into a ranging error of $c\,\delta_{i\ell}$; motivating PHASOR's tight synchronization target.

We evaluate localization across 3 datasets, described in figure 5. In each dataset a small wheeled robot with a planar LiDAR is used to collect ground-truth position estimates. Wi-Fi packets are sent from an ESP32 mounted to the robot and received at every node. Figure 7 plots a CDF of X-Y localization error for each dataset. PHASOR obtains a 2.48 meter median error in the best-case scenario, with performance degrading under non-line-of-sight conditions, as the propagation delay no longer accurately reflects the distance.

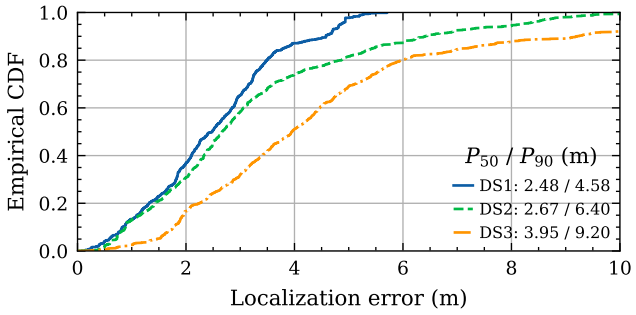


Fig. 7: CDF of X-Y positioning error across 3 datasets. Median and 90th percentile errors are reported in the legend.

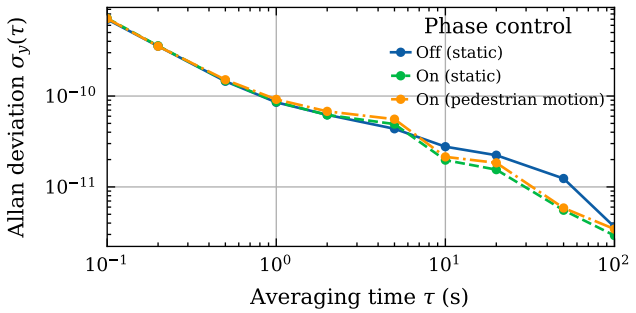


Fig. 8: 10MHz Allan deviation plots with/without the phase controller (§IV-D) enabled, tested under line-of-sight conditions. Since the phase controller assumes channel stability, we test the effect of typical indoor dynamic multipath by having a human walk back and forth across the link over the 10 minute experiment period, obstructing the line-of-sight. Disabling phase control results in larger deviations in between FTM measurement exchanges (1-10 second region), while not significantly affecting phase noise (<1 second region.)

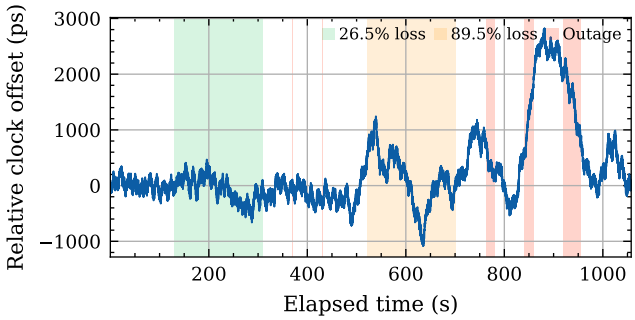


Fig. 9: 10MHz phase drift measured at the Sentinel under various communication drops. The system is robust to small drop rates, while a 90% packet loss rate causes large deviations around $1ns$, and occasional hard outages cause significant excursions.

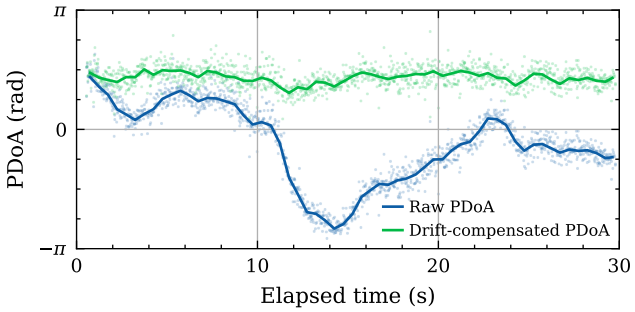


Fig. 10: Phase difference on arrival from a static transmitter measured at two PHASOR nodes. The raw measurement without LO phase removal from §IV-E (blue) drifts due to the action of the controller; estimating this drift by combining the bidirectional phase offset between nodes cancels it effectively (green).

C. Ablations

1) Disabling Phase Control: To evaluate the impact of the LO phase measurements on the synchronization performance, in figure 8 we perform the 10MHz offset experiment in §VI-A with phase synchronization disabled.

2) Robustness to Communication Drops: In figure 9 we demonstrate the 10MHz phase offset over time under several adverse channel contention effects. Occasional packet drops are induced in software at a 25% and 90% rate, as well as occasional outages. PHASOR can maintain nanosecond-level sync even with most FTM exchanges failing, but an extended outage causes significant phase deviation as the oscillator begins to drift.

3) Static PDoA Stability: PHASOR provides a full phase-coherent distributed aperture, allowing for sensing techniques that take into account channel phase as well as delay. We demonstrate this by plotting relative channel phase between two PHASOR nodes over time for a static transmitter. The phase correction in section IV-E ensures that the phase difference remains constant over a long period. In this work, we do not fully exploit this phase offset, as using it for localization requires centimeter-accurate knowledge of the position of each node to much less than 1 wavelength of the carrier frequency.

VII. Discussion & Future Work

PHASOR demonstrates that precise time and phase synchronization, previously available only through wired timing backbones or specialized radios, can be obtained entirely over the air on commodity Wi-Fi hardware. This synchronization infrastructure enables new sensing applications such as reverse-TDoA localization of unmodified transmitters, requiring no cooperation from the device being located. We identify several areas of interest for future research:

A. Distributed Control

In this work, one node is designated the leader and all followers steer to it. A natural extension is fully distributed control, in which nodes run pairwise FTM exchanges with their neighbors and estimate a consensus clock state, in the style of distributed clock-servo protocols. This removes the single point of failure and allows the synchronization graph,

rather than a fixed star, to follow deployment geometry. In a distributed setting every node is simultaneously steered and steering, so the estimator must account for coupled dynamics across the graph.

B. Exploiting Phase Synchronization

PHASOR's fine loop aligns receiver LOs to within a fraction of a carrier cycle, effectively creating a distributed aperture whose elements are separated by meters rather than half-wavelengths. Fully exploiting this aperture is nontrivial. Phase-difference-of-arrival (PDoA) techniques resolve angle, and hence position, from inter-receiver carrier phase, but phase is observed only modulo 2π; unambiguous operation requires element spacing below $\lambda/2 \approx 6\,\text{cm}$ at 2.4 GHz. Real distributed apertures exceed this significantly, so PDoA observations are heavily aliased. Resolving this ambiguity requires significant future work.

REFERENCES

[1] N. Jadhav, W. Wang, D. Zhang, O. Khatib, S. Kumar, and S. Gil, "WSR: A WiFi sensor for collaborative robotics," *arXiv preprint arXiv:2012.04174*, 2020.

[2] R. Liu, S. H. Marakkalage, M. Padmal, T. Shaganan, C. Yuen, Y. L. Guan, and U.-X. Tan, "Collaborative SLAM based on Wifi fingerprint similarity and motion information," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1826–1840, 2019.

[3] A. Alanwar, H. Ferraz, K. Hsieh, R. Thazhath, P. Martin, J. Hespanha, and M. Srivastava, "D-SLATS: Distributed simultaneous localization and time synchronization," in *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2017, pp. 1–10.

[4] S. Li, M. Hedley, K. Bengston, M. Johnson, D. Humphrey, A. Kajan, and N. Bhaskar, "Tdoa-based passive localization of standard wifi devices," in *2018 Ubiquitous Positioning, Indoor Navigation and Location-Based Services (UPINLBS)*, 2018, pp. 1–5.

[5] T. Xie, H. Jiang, X. Zhao, and C. Zhang, "A Wi-Fi-based wireless indoor position sensing system with multipath interference mitigation," *Sensors*, vol. 19, no. 18, 2019.

[6] S. A. Golden and S. S. Bateman, "Sensor measurements for Wi-Fi location with emphasis on time-of-arrival ranging," *IEEE Transactions on Mobile Computing*, vol. 6, no. 10, pp. 1185–1198, 2007.

[7] A. Mahmood, R. Exel, H. Trsek, and T. Sauter, "Clock synchronization over IEEE 802.11—a survey of methodologies and protocols," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 907–922, 2017.

[8] P. Chen and Z. Yang, "Understanding precision time protocol in today's Wi-Fi networks: A measurement study," in *USENIX Annual Technical Conference (USENIX ATC)*, 2021, pp. 597–610.

[9] T. Hao, R. Zhou, G. Xing, M. W. Mutka, and J. Chen, "WizSync: Exploiting Wi-Fi infrastructure for clock synchronization in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 13, no. 6, pp. 1379–1392, 2014.

[10] K. Alemdar, D. Varshney, S. Mohanti, U. Muncuk, and K. Chowdhury, "RFClock: Timing, phase and frequency synchronization for distributed wireless networks," in *ACM Annual International Conference on Mobile Computing and Networking (ACM MobiCom)*, 2021, pp. 15–27.

[11] A. Dongare, P. Lazik, N. Rajagopal, and A. Rowe, "Pulsar: A wireless propagation-aware clock synchronization platform," in *2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*. IEEE, 2017, pp. 283–292.

[12] H. V. Balan, R. Rogalin, A. Michaloliakos, K. Psounis, and G. Caire, "AirSync: Enabling distributed multiuser MIMO with full spatial multiplexing," *IEEE/ACM Transactions on Networking*, vol. 21, no. 6, pp. 1681–1695, 2013.

[13] F. Euchner, M. Gauger, S. Dörner, and S. ten Brink, "A distributed massive MIMO channel sounder for big CSI data-driven machine learning," in *International ITG Workshop on Smart Antennas (WSA)*, 2021.

[14] M. Sandra, C. Nelson, X. Li, X. Cai, F. Tufvesson, and A. J. Johansson, "A wideband distributed massive mimo channel sounder for communication and sensing," *IEEE Transactions on Antennas and Propagation*, vol. 73, no. 4, pp. 2074–2085, 2025.

[15] G. Callebaut, J. V. Mulders, G. Ottoy, D. Delabie, B. Cox, N. Stevens, and L. V. d. Perre, "Techtile — open 6G R&D testbed for communication, positioning, sensing, WPT and federated learning," in *Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, 2022, pp. 417–422.

[16] A. Colpaert, S. De Bast, R. Beerten, A. P. Guevara, Z. Cui, and S. Pollin, "Massive MIMO channel measurement data set for localization and communication," *IEEE Communications Magazine*, vol. 61, no. 9, pp. 114–120, 2023.

[17] J. Tiemann, Y. Elmasry, L. Koring, and C. Wietfeld, "ATLAS FaST: Fast and simple scheduled TDoA for reliable ultra-wideband localization," in *IEEE International Conference on Robotics and Automation (IEEE ICRA)*, 2019, pp. 2554–2560.

[18] J. Friedrich, J. Tiemann, and C. Wietfeld, "Accurate multi-zone UWB TDoA localization utilizing cascaded wireless clock synchronization," in *IEEE International Conference on Indoor Positioning and Indoor Navigation (IEEE IPIN)*, 2021, pp. 1–8.

[19] F. Euchner and S. T. Brink, "ESPARGOS: Phase-coherent WiFi CSI datasets for wireless sensing research," in *2024 Kleinheubach Conference*, 2024, pp. 1–4.

[20] Espressif Systems, "ESP-CSI: Applications based on Wi-Fi CSI (channel state information)," https://github.com/espressif/esp-csi, 2026, accessed: 2026-07-22.

[21] S. M. Hernandez and E. Bulut, "Performing wifi sensing with off-the-shelf smartphones," in *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, 2020, pp. 1–3.

[22] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: gathering 802.11n traces with channel state information," *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 1, p. 53, Jan. 2011. [Online]. Available: https://doi.org/10.1145/1925861.1925870

[23] Y. Xie, Z. Li, and M. Li, "Precise power delay profiling with commodity wi-fi," *IEEE Transactions on Mobile Computing*, vol. 18, no. 6, pp. 1342–1355, 2019.

[24] F. Gringoli, M. Schulz, J. Link, and M. Hollick, "Free your csi: A channel state information extraction platform for modern wi-fi chipsets," in *Proceedings of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*, ser. WiNTECH '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 21–28. [Online]. Available: https://doi.org/10.1145/3349623.3355477

[25] R. Kim, T. Ha, H. Lim, and D. Jung, "Tdoa localization for wireless networks with imperfect clock synchronization," in *The International Conference on Information Networking 2014 (ICOIN2014)*, 2014, pp. 417–421.

[26] T. Wang, H. Xiong, H. Ding, and L. Zheng, "Tdoa-based joint synchronization and localization algorithm for asynchronous wireless sensor networks," *IEEE Transactions on Communications*, vol. 68, no. 5, pp. 3107–3124, 2020.

[27] K. Kosek-Szott, S. Szott, W. Ciezobka, M. Wojnar, K. Rusek, and J. Segev, "Indoor positioning with wi-fi location: A survey of ieee 802.11mc/az/bk fine timing measurement research," 2025. [Online]. Available: https://arxiv.org/abs/2509.03901

[28] L. Banin, O. Bar-Shalom, N. Dvorecki, and Y. Amizur, "Scalable wi-fi client self-positioning using cooperative ftm-sensors," *IEEE Transactions on Instrumentation and Measurement*, vol. 68, no. 10, pp. 3686–3698, 2019.

[29] I. Martin-Escalona and E. Zola, "Passive round-trip-time positioning in dense ieee 802.11 networks," *Electronics*, vol. 9, no. 8, 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/8/1193

[30] A. Kulkarni and A. Lim, "Preliminary study on indoor localization using smartphone-based ieee 802.11 mc," in *Proceedings of the 15th International Conference on emerging Networking EXperiments and Technologies*, 2019, pp. 43–44.

[31] C. Zucca and P. Tavella, "The clock model and its relationship with the allan and related variances," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 52, no. 2, pp. 289–296, 2005.

[32] Calnex, "Sentinel: Verify static and dynamic time, phase and frequency synchronization performance on your packet network," https://calnexsol.com/products/sentinel/, 2024, accessed: 2025-01-31.