



Budapest University of Technology and Economics
Doctoral School of Informatics
Department of Telecommunications and Artificial Intelligence

Time synchronization in packet switching networks

PH.D. DISSERTATION

Ph.D. candidate
Gergely Hollósi

Supervisor
Pál Varga, Ph.D.

February 22, 2025

„Reality is frequently inaccurate.”

– Douglas Adams, *The Restaurant at the End of the Universe*

Abstract

Time synchronization holds an essential role in computer science, particularly in methods that offer precise time alignment down to nanosecond levels across packet-switched networks. Fields such as robotics, advanced communication networks like 5G/6G, and high-fidelity localization systems heavily rely on precise time synchronization. Among the technologies facilitating accurate timestamping, this research primarily focuses on the widely utilized Precision Time Protocol and ultra-wideband technology, both encountering similar challenges regarding synchronization and synthonization. Addressing clock state estimation within packet-switched networks, optimal estimation deals with the inherently stochastic nature of packet delay variation, typically requiring a priori specification.

This study introduces two methods for achieving adaptive estimation by online estimation of the packet delay variation: employing an expectation-maximization approach for ultra-wideband frequency synchronization, ensuring convergence to an optimal solution, and utilizing the Adaptive Kalman filter in IEEE 1588 systems, offering a computationally efficient approximation to the optimal solution. While optimizing estimation yields superior results, utilizing enhanced measurement models can further boost performance. This work shows that leveraging a higher effective data rate in ultra-wideband communication can increase the number of measurements available to the estimator. Lastly, ultra-wideband technology is brought together with the Precision Time Protocol through specific use cases, necessitating an implementation method to achieve synchronization of standard PTP clocks via ultra-wideband communication, including resolving the issue of chip-level clock domain alteration. All proposed solutions in this study underwent evaluation and validation using real and simulated measurements.

Acknowledgement of AI Assistance

I acknowledge the use of ChatGPT¹ as a tool for rephrasing certain sections of this dissertation to improve clarity and coherence. All ideas, arguments, and conclusions presented are entirely my own, and the rephrased content faithfully reflects my original thoughts and intentions. The only part created entirely by ChatGPT is the Acknowledgement.

¹<https://chatgpt.com/>

Acknowledgement

Thanks for all your support, kindness, and generosity, which have made a profound impact on my life and filled it with warmth and gratitude.

– ChatGPT

Contents

1	Introduction	6
1.1	Time synchronization and requirements	7
1.2	Overview of the dissertation	8
2	Methods and related works	10
2.1	Bayesian estimation	10
2.1.1	General method	10
2.1.2	Bayesian estimation in Markov processes	12
2.2	Digital clocks and models	14
2.2.1	Ideal oscillator	16
2.2.2	Simple skew model	16
2.2.3	State space probabilistic models	17
2.2.4	Measuring frequency stability	19
2.3	Clock synchronization	21
2.3.1	IEEE 1588 standard	23
2.3.2	Controllable and virtual clocks	26
2.3.3	Clock state estimation algorithms	28
2.3.4	Precision and accuracy	29
2.4	Ultra wide band communication	29
3	Efficient ultra wide band measurement system design	32
3.1	Increasing the effective data rate for enhancing measurement frequency	33
3.1.1	Proposed solution	34
3.1.2	Analysis of ranging and localization rate	37
3.1.3	Analysis of ranging error	39
3.2	Evaluation	41
3.3	Thesis summary	45
4	Synchronizing PTP clocks using UWB	46
4.1	Use cases	48
4.2	Synchronizing local clock domains	48
4.3	Synchronizing local PTP clock using ultra wide band radio	50
4.3.1	The hardware PTP clock	52
4.3.2	Synchronization with the DW1000 chip	52
4.3.3	Timestamping errors	55

4.3.4	Results	56
4.4	Thesis summary	59
5	Measurement distribution adaptive synthonization in UWB communication systems	60
5.1	Modeling the distribution of receive timestamps	61
5.1.1	Channel model	61
5.1.2	Pulse shaping	63
5.1.3	Noise	63
5.1.4	Timestamping	64
5.1.5	Results	64
5.2	Measuring the distribution of receive timestamps in UWB systems	67
5.3	Estimating measurement noise of the distribution of the received timestamps	73
5.3.1	Measurement method	73
5.3.2	Linear filter model	75
5.3.3	Estimating the measurement noise	76
5.3.4	Results	77
5.3.5	Conclusion	83
5.4	Thesis summary	85
6	Precise time synchronization in IEEE 802.1AS systems	86
6.1	Adaptive estimation of clock state in PTP clock servo	87
6.1.1	Adaptive Kalman filter	88
6.1.2	Time synchronization model	90
6.2	Evaluation of proposed method	92
6.2.1	Remarks on computational complexity	98
6.3	Thesis summary	102
7	Applicability of the results and future outlook	103
8	Summary	105
	Bibliography	107
	Publications	119

Acronyms

- AKF** Adaptive Kalman Filter. [7](#), [85](#), [88](#)
- BFGS** Broyden–Fletcher–Goldfarb–Shanno algorithm. [74](#), [85](#)
- CIR** channel impulse response. [45](#), [68](#)
- DS-TWR** double sided two-way ranging. [30](#)
- EM** Expectation-Maximization. [74](#)
- FIR** Finite Impulse Response. [54](#)
- FTM** fine timing measurement. [45](#)
- GNSS** Global Navigation Satellite System. [7](#)
- GPS** Global Positioning System. [44](#), [46](#)
- HRP UWB** high-rate pulse repetition frequency UWB. [28](#)
- IIoT** Industrial Internet of Things. [6](#)
- IIR** Infinite Impulse Response. [54](#)
- IMM** Interacting Multiple Models. [85](#)
- KL** Kullback-Leibler divergence. [10](#)
- LED** Leading Edge Detector. [62](#), [69](#)
- LQR** Linear Quadratic Regulators. [25](#)
- LRP UWB** low-rate pulse UWB. [28](#)
- LSB** least significant bit. [28](#)
- MAC** medium access control. [48](#)

MAP maximum a posteriori. [10](#)

ML maximum likelihood. [10](#)

NLOS non-line of sight. [30](#), [53](#), [58](#), [60](#)

NTP Network Time Protocol. [6](#)

OCXO Oven-controlled crystal oscillator. [18](#), [47](#)

OSTR one-shot timebase reset. [50](#)

OSTS one-shot transmit synchronization mode. [50](#), [52](#)

PDP power delay profile. [60](#)

PDV packet delay variation. [7](#), [26](#), [86](#)

PHC PTP hardware clock. [50](#)

PHR PHY header. [31](#)

PI Proportional Integral. [24](#), [54](#)

PLL Phased-locked loop. [21](#), [48](#)

PPS pulse per second. [46](#), [55](#)

PRF pulse repetition frequency. [31](#), [35](#)

PSDU PHY Service Data Unit. [31](#)

PTP Precision Time Protocol. [6](#), [21](#), [24](#), [44](#), [46](#), [85](#)

PTP-LP PTP linear programming. [26](#)

RRC root raised cosine. [64](#)

RTS Rauch-Tung-Striebel smoother. [75](#)

SGD Stochastic Gradient Descent. [74](#)

SHR Synchronization header. [31](#)

SKM Simple Skew Model. [15](#), [28](#), [37](#)

SoC system on a chip. [46](#), [50](#)

SV Saleh-Valenzuela model. [59](#)

SyncE Synchronous Ethernet. [23](#)

TCXO Temperature-compensated crystal oscillator. [18](#), [47](#), [48](#), [65](#)

TDoA time difference of arrival. [30](#), [58](#)

ToA time of arrival. [30](#)

ToF time-of-flight. [28](#)

TSN Time-Sensitive Networking. [6](#), [23](#), [45](#)

TWR two-way ranging. [30](#), [35](#), [39](#), [40](#)

UDP User Datagram Protocol. [21](#)

UWB Ultra-Wideband. [6](#), [27](#), [30](#), [35](#), [44](#), [59](#)

WiFi Wireless Fidelity. [45](#)

WLAN Wireless Local Area Network. [44](#)

WSN Wireless Sensor Networks. [6](#)

1

Introduction

The need for synchronizing clocks within computer networks has a lengthy history, but contemporary time synchronous networks present novel challenges to expected accuracy. Accurate timekeeping is gaining increased attention in practical [Industrial Internet of Things \(IIoT\)](#) applications and various use-cases, such as real-time scenarios like nuclear fusion control, mobile communication (5G/6G), substation automation, and modern manufacturing plants [1, 2]. The well-established [Network Time Protocol \(NTP\)](#) was originally designed for time transfer in information technology services, with accuracy in the millisecond range. However, the emergence of [Time-Sensitive Networking \(TSN\)](#) demands higher synchronization accuracy, reaching into the order of microseconds or, as nowadays, nanoseconds [2, 3]. Time-sensitive networking finds application in various areas, including Industrial Internet-of-Things [4–6], automotive embedded systems [7], and mobile networking [8–10]. Additionally, the recent IEEE 1588-2019 standard [11], commonly known as the [Precision Time Protocol \(PTP\)](#), addresses the challenge of sub-microsecond accurate clock synchronization over communication networks.

Furthermore, high-accuracy time synchronization is essential in indoor and outdoor localization systems, where clocks in base stations (referred to as anchors) need synchronization with each other or a global logical clock. For instance, techniques like time difference of arrival in 5G networks [12, 13] or [Ultra-Wideband \(UWB\)](#) systems [14–17] require synchronization of frequency drifts at the nanosecond level. Localization is a critical mechanism for determining the locations of data sources in [Wireless Sensor Networks \(WSN\)](#), autonomous robots, or the Industrial Internet of Things [18]. A prime example of localization-focused technologies, the UWB technology offers robustness against narrowband interference, small-scale fading, multi-path immunity, and strong obstacle penetration during communication and distance measurements. To achieve centimeter-

level positioning, UWB utilizes impulse radio signals with pulse durations in the order of fractions of nanoseconds [19–21].

In the realm of accurate and precise time synchronization, three prominent technologies come to mind: the [Global Navigation Satellite System \(GNSS\)](#) (e.g. [Global Positioning System \(GPS\)](#) [22]), Precision Time Protocol, and Ultra-Wideband technology. The interconnection of time synchronization and localization is undeniable, as most positioning methods favor time-of-flight or time-difference-of-arrival techniques, requiring accurate time measurements in transceivers. This dissertation focuses on the precise time synchronization in packet-switched networks – precise in aiming for high precision (low variance) beyond accuracy and packet-switched because the findings can generally be applied to time synchronization of digital clocks through any packet-switched communication systems.

1.1 Time synchronization and requirements

In computer science, time synchronization in packet-switched networks began with the [Network Time Protocol \(NTP\)](#), introduced in 1985 by David L. Mills [23]. NTP was designed to synchronize clocks of computer systems over variable-latency networks, achieving millisecond-level accuracy. It operates on a hierarchical system of time sources, known as strata, to disseminate time information from primary reference clocks to clients across the network. As the demand for higher precision grew, especially in industrial and scientific applications, the [Precision Time Protocol \(PTP\)](#) was standardized as IEEE 1588 in 2002 [24]. PTP offers sub-microsecond accuracy by minimizing the variability in message delivery times, making it suitable for local area networks where such precision is essential. Building upon PTP, the White Rabbit project was initiated in 2009 at CERN. This project aimed to achieve sub-nanosecond synchronization over Ethernet networks. White Rabbit [3] combines PTP with Synchronous Ethernet (SyncE) and incorporates precise calibration techniques to account for link asymmetries and environmental variations, resulting in unprecedented synchronization accuracy.

Different industries and applications have specific timing accuracy requirements to ensure optimal performance. Table 1.1 summarizes the timing accuracy requirements of different standards over different application areas. The currently used protocols with their timing accuracies can be seen in Table 1.2. The strictest requirements are found in 5G systems [25], where synchronization must be maintained even within 900 nanoseconds. Other high-precision applications, such as industrial automation, audio/video broadcasting (AES67 [26]), and the IEEE 802.1AS (gPTP) [27] standard for time-sensitive networking, demand sub-microsecond accuracy. Similarly, the IEEE C37.238 standard [28] for smart grids requires synchronization within 1 microseconds to ensure precise control and monitoring of power systems. The IEEE 1588 power profile for electrical grid applications holds accuracy within 4 microseconds. Financial trading regulations (MiFID II) allow for a slightly more relaxed limit of 100 microseconds to ensure fair and reliable market transactions. These requirements highlight the critical

Table 1.1: Timing accuracy requirements in different standards and application areas

Standard	Required Accuracy
Financial Trading (MiFID II) [29]	$\leq 100 \mu s$
Electrical Grid (Smart Grid) [30]	$4 \mu s$
IEEE 802.1AS (gPTP) [27]	$\leq 1 \mu s$
Industrial Automation [31]	$\leq 1 \mu s$
Audio/Video Broadcasting (AES67) [26]	$\leq 1 \mu s$
Smart Grid (IEEE C37.238) [28]	$\leq 1 \mu s$
3GPP TS 22.104 (5G Systems) [25]	$\leq 900 ns$

Table 1.2: Timing accuracy of existing time synchronization protocols

Protocol	Accuracy	Description
Network Time Protocol (RFC 5905) [23]	1–10 ms	General-purpose time synchronization for internet-based systems
Precision Time Protocol (IEEE 1588) [11]	100 ns– $1 \mu s$	Industrial automation, telecommunications, and power grids
Synchronous Ethernet (SyncE) [32]	Only frequency synchronization	Frequency synchronization through Ethernet physical layer
White Rabbit [3]	$< 1 ns$	High-energy physics and scientific research

role of precise time synchronization in modern packet-switched networks, where even microsecond-level deviations can impact performance and reliability.

However, it is important to distinguish between time synchronization and the actual use of synchronized time. Even if a clock meets the required accuracy, it does not necessarily guarantee that actions taken based on that clock will comply with strict timing requirements. This work focuses only on synchronizing two digital clocks, but it is crucial to remember that executing actions based on synchronized time requires careful system design.

1.2 Overview of the dissertation

The dissertation discusses three different topics separated into four thesis groups, all topics concentrating on clock synchronization issues. Chapter 3 presents a state-of-the-art time division method for increasing the number of time measurements in ultra wide band networks, resulting in more information about the clock state of the nodes in the UWB network, facilitating more accurate ranging and time synchronization compared

to existing solutions. The chapter also presents that the framing structure performs similar ranging metrics as the method of the IEEE 802.15.4 [14] standard.

The connection between the high accuracy requirements of the time measurement in wireless localization (e.g. ranging) and the time synchronization solutions raises the question: is UWB applicable to synchronize controllable PTP clocks? Chapter 4 discusses the topic by proposing a specific implementation based on the well-known DW1000 chipset, pointing out the problem of chip-level clock domain change.

Chapter 5 handles the topical problem of unknown propagation delay variations over wireless channels, which decline the performance of clock synthonization (frequency synchronization) algorithms. While a couple of methods are proposed for the problem, the linearity of the dynamic and measurement models makes it possible to apply Kalman-filtering, resulting in superior performance – however, Kalman-filtering requires the estimate of the propagation delay variance to be the optimal estimator. The chapter lends a method from machine learning to estimate parameters in latent space problems, which results in the optimal estimate of both the states and the measurement variance.

Measurement noise variance, which is mostly referred to as [packet delay variation \(PDV\)](#) in wired networks, is also a problem in the Precision Time Protocol. While numerous technical solutions have been proposed and implemented – like transparent clocks or the White Rabbit Project [3] – to overcome the unknown delays of the synchronizing packets over the network, statistical methods can always help to make optimal estimation of the clock state by inferring the measurement noise on-the-fly. Chapter 6 proposes the [Adaptive Kalman Filter \(AKF\)](#) to adaptively estimate the measurement noise, by applying a measurement model with time-invariant measurement noise covariance matrix. The method is shown to supersede existing state-of-the-art solutions for estimating the clock state.

To establish the basis for the discussion of the algorithms proposed in the dissertation, Chapter 2 presents the most important methods and methodologies appearing in this work.

2

Methods and related works

2.1 Bayesian estimation

Bayesian estimation is used in a variety of applications, e.g. in tracking applications [33], image processing [34], signal processing [35], telecommunications [36], and also, Bayesian estimation is the backbone of machine learning and artificial intelligence [37]. Bayesian estimation plays a crucial role in this work; therefore, its main aspects are discussed. For a more in-depth understanding of Bayesian methods, particularly for filtering purposes, see [38].

2.1.1 General method

A commonly recurrent issue in engineering is the so-called parameter estimation problem. Suppose, we have a $\boldsymbol{\theta} = \{\theta_1, \theta_2, \dots, \theta_D\}^T$ vector with D dimensions, called the *parameters* or *parameter vector*. Suppose also, we know the conditional distribution of $p(\mathbf{y}_{1:T}|\boldsymbol{\theta})$, where $\mathbf{y}_{1:T} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T\}$ are the *measurements* as a time series from timestep 1 to T . The probability density function or the probability mass function (in the case of discrete variables) are both denoted by $p(\cdot)$. The Bayesian estimation handles $\boldsymbol{\theta}$ as a multivariate random variable and finds the distribution for the parameters using the Bayes rule, assuming a *prior* distribution $p(\boldsymbol{\theta})$ for the parameters:

$$p(\boldsymbol{\theta}|\mathbf{y}_{1:T}) = \frac{p(\mathbf{y}_{1:T}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{y}_{1:T})} = \frac{p(\mathbf{y}_{1:T}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{y}_{1:T}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}$$

Please note that the integral without limits means here that the expression shall be integrated with respect to the multidimensional $\boldsymbol{\theta}$ by each axis and for each possible interval – practically for the whole sample space, i.e. from $-\infty$ to ∞ by each axis.

Using the Bayes rule, the result is a probability distribution for the unknown parameters, however, if a point estimate is required, the maximum value of the density (the mode) can be considered, which results in a *maximum a posteriori* estimation (MAP):

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} = \arg \max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{y}_{1:T}) = \arg \max_{\boldsymbol{\theta}} \frac{p(\mathbf{y}_{1:T}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{y}_{1:T})}$$

Given certain measurements $\mathbf{y}_{1:T}$, $p(\mathbf{y}_{1:T})$ can be considered as a constant, so

$$p(\boldsymbol{\theta}|\mathbf{y}_{1:T}) \propto p(\mathbf{y}_{1:T}|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

resulting in the estimation of

$$\hat{\boldsymbol{\theta}}_{\text{MAP}} = \arg \max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathbf{y}_{1:T}) = \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}_{1:T}|\boldsymbol{\theta})p(\boldsymbol{\theta})$$

Using a uninformative, or uniform prior (i.e. $p(\boldsymbol{\theta}) \propto 1$), the method falls back to the classical *maximum likelihood* estimation (ML):

$$\hat{\boldsymbol{\theta}}_{\text{ML}} = \arg \max_{\boldsymbol{\theta}} p(\mathbf{y}_{1:T}|\boldsymbol{\theta})$$

In typical practical applications, instead of maximizing the likelihood itself, the natural logarithm of the likelihood (called the *log-likelihood*) is applied. Beyond the numerical and the information theory related reasons (i.e. the evaluation of the [Kullback-Leibler divergence \(KL\)](#)), this makes it possible to decouple the observations, provided the observations are conditionally independent:

$$p(\mathbf{y}_{1:T}|\boldsymbol{\theta}) = \prod_{i=1}^T p(\mathbf{y}_i|\boldsymbol{\theta})$$

in which case the log-likelihood is the sum of the likelihoods of the observations:

$$\ln p(\mathbf{y}_{1:T}|\boldsymbol{\theta}) = \sum_{i=1}^T \ln p(\mathbf{y}_i|\boldsymbol{\theta})$$

Using the log-likelihood in the optimization, the maximum likelihood estimation is the $\hat{\boldsymbol{\theta}}_{\text{ML}}$, which satisfies the equation

$$\sum_{i=1}^T \frac{1}{p(\mathbf{y}_i|\hat{\boldsymbol{\theta}}_{\text{ML}})} \left. \frac{\partial p(\mathbf{y}_i|\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{\text{ML}}} = 0$$

Please note that throughout the dissertation, “optimal” always refers to, “optimal in a maximum likelihood sense” unless otherwise noted.

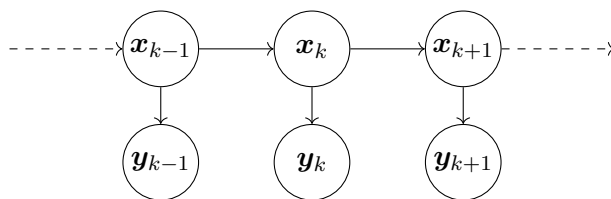


Figure 2.1: First order Markov process

2.1.2 Bayesian estimation in Markov processes

Numerous engineering problems can be modeled as latent state problems, where the $\mathbf{x}_{1:T}$ *latent states* can be observed only through the $\mathbf{y}_{1:T}$ *measurements* depending on the latent states. Specifying a latent state process requires the $p(\mathbf{x}_k | \mathbf{x}_0, \dots, \mathbf{x}_{k-1}, \mathbf{x}_{k+1}, \dots, \mathbf{x}_T, \mathbf{y}_{1:T})$ latent space conditional distributions, the $p(\mathbf{y}_k | \mathbf{x}_{0:T}, \mathbf{y}_0, \dots, \mathbf{y}_{k-1}, \mathbf{y}_{k+1}, \dots, \mathbf{y}_T)$ measurement conditional distributions and the $p(\mathbf{x}_0)$ prior distribution to be defined for all $k = 1, \dots, T$. The k is typically called *time step*.

Special cases of the latent space problems are the first-order Markov processes, as depicted by the acyclic directed probability graph in Figure 2.1. In a first order Markov process, the \mathbf{x}_k *latent state* depends only on the previous latent space, called the *dynamic model*, i.e.

$$p(\mathbf{x}_k | \mathbf{x}_{1:T}, \mathbf{y}_{1:T}) = p(\mathbf{x}_k | \mathbf{x}_{k-1}) \quad (2.1)$$

and the measurements depend only on the latent space at that time step, called the *measurement model*, i.e.

$$p(\mathbf{y}_k | \mathbf{x}_{1:T}, \mathbf{y}_{1:T}) = p(\mathbf{y}_k | \mathbf{x}_k) \quad (2.2)$$

The constraints on the dynamic and measurement model make the factorization of the joint distribution possible, i.e.

$$p(\mathbf{x}_{0:T}) = p(\mathbf{x}_0) \prod_{k=0}^{T-1} p(\mathbf{x}_{k+1} | \mathbf{x}_k) \quad (2.3a)$$

$$p(\mathbf{y}_{1:T} | \mathbf{x}_{0:T}) = \prod_{k=1}^T p(\mathbf{y}_k | \mathbf{x}_k) \quad (2.3b)$$

Bayesian filtering

Typically, one can be interested in estimating the actual state, knowing the previous state(s) and the actual measurement, to estimate the $p(\mathbf{x}_k | \mathbf{x}_{0:k-1}, \mathbf{y}_{1:k})$. The method is called *Bayesian filtering*, and has three basic steps, as

Initialization The recursive filtering starts from $p(\mathbf{x}_0)$

State prediction Knowing $p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1})$, $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ can be calculated with the well-know Chapman-Kolmogorov equation:

$$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1}) d\mathbf{x}_{k-1}$$

State update Given a new measurement \mathbf{y}_k , update the state estimation, which yields

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \frac{p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1})}{\int p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) d\mathbf{x}_k}$$

A special case of Bayesian filtering is the linear Gaussian problem, commonly called *Kalman-filter* [39]. In Kalman filtering, both the dynamic and the measurement models are linear and have Gaussian noises. Let

$$\begin{aligned} p(\mathbf{x}_k|\mathbf{x}_{k-1}) &= \mathcal{N}(\mathbf{x}_k|A_k\mathbf{x}_{k-1}, \Sigma_k) \\ p(\mathbf{y}_k|\mathbf{x}_k) &= \mathcal{N}(\mathbf{y}_k|H_k\mathbf{x}_k, \Omega) \end{aligned} \quad (2.4)$$

where A_k is the state-transition matrix, H_k is the observation matrix, Σ_k is the process noise covariance and Ω is the observation noise covariance. Also, let $\mathcal{N}(\cdot|\mathbf{m}, \Sigma)$ denote the probability density function of a multivariate Gaussian distribution with expected value \mathbf{m} and covariance Σ , i.e.

$$\mathcal{N}(\mathbf{x}|\mathbf{m}, \Sigma) = \frac{1}{\sqrt{(2\pi)^D|\Sigma|}} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{m})^T\Sigma^{-1}(\mathbf{x}-\mathbf{m})}$$

where D is the dimension of the multivariate distribution, and $|\cdot|$ means the determinant. It can be proven, that $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ is also Gaussian, i.e. $p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \mathcal{N}(\mathbf{x}_k|\mathbf{m}_k, P_k)$, where \mathbf{m}_k and P_k can be calculated with the Kalman-filtering equations

$$\begin{aligned} \mathbf{m}_k^- &= A_{k-1}\mathbf{m}_{k-1} \\ P_k^- &= A_{k-1}P_{k-1}A_{k-1}^T + \Sigma_{k-1} \end{aligned} \quad (2.5)$$

$$\begin{aligned} K_k &= P_k^- H_k^T (H_k P_k^- H_k^T + \Omega)^{-1} \\ \mathbf{m}_k &= \mathbf{m}_k^- + K_k(\mathbf{y}_k - H_k \mathbf{x}_k^-) \\ P_k &= (I - K_k H_k) P_k^- \end{aligned} \quad (2.6)$$

The matrix I denotes the appropriate sized identity matrix.

Bayesian smoothing

Based on the results of a Bayesian filtering, *Bayesian smoothing* calculates the distribution of a state \mathbf{x}_k conditioned on the measurements $\mathbf{y}_{1:T}$, where $T > k$, i.e.

$$p(\mathbf{x}_k|\mathbf{y}_{1:T})$$

The basic equations for Bayesian smoothing use the dynamic model presented in (2.1), and also the $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ distributions resulted from the filtering process. The basic equation for smoothing is

$$p(\mathbf{x}_k|\mathbf{y}_{1:T}) = p(\mathbf{x}_k|\mathbf{y}_{1:k}) \int \frac{p(\mathbf{x}_{k+1}|\mathbf{x}_k)p(\mathbf{x}_{k+1}|\mathbf{y}_{1:T})}{p(\mathbf{x}_{k+1}|\mathbf{y}_{1:k})} d\mathbf{x}_{k+1} \quad (2.7)$$

Please note that the distribution $p(\mathbf{x}_{k+1}|\mathbf{y}_{1:k})$ in the denominator can be calculated using the Chapman-Kolmogorov equation. The smoothing shall be done backward, starting from $p(\mathbf{x}_T|\mathbf{y}_{1:T})$. Thus, the smoother is sometimes called as *backward filter*.

A special case of Bayesian smoothing is the linear Gaussian problem, commonly called *Rauch-Tung-Striebel smoother* [40]. For the linear model presented in (2.4), the backward filter results in $p(\mathbf{x}_k|\mathbf{y}_{1:T}) = \mathcal{N}(\mathbf{x}_k|\mathbf{m}_k^s, P_k^s)$, with

$$\begin{aligned} \mathbf{m}_{k+1}^- &= A_k \mathbf{m}_k \\ P_{k+1}^- &= A_k P_k A_k^T + \Sigma_k \\ G_k &= P_k A_k^T (P_{k+1}^-)^{-1} \\ \mathbf{m}_k^s &= \mathbf{m}_k + G_k (\mathbf{m}_{k+1}^s - \mathbf{m}_{k+1}^-) \\ P_k^s &= P_k + G_k (P_{k+1}^s - P_{k+1}^-) G_k^T \end{aligned} \quad (2.8)$$

2.2 Digital clocks and models

Wide variety of applications use digital clocks to represent the wall-clock time in digital systems. For defining the concept of *clock* in computer science, here is a quote from the IEEE 1588-2019 standard [11]:

[A clock in computer science is] „a device that can provide a measurement of the passage of time since a defined epoch. A clock provides time at desired moments of the timescale it maintains. Time is obtained either:

Physically in this type of clock, the time is modeled using a clock signal and a time counter that is driven by the clock signal;

Mathematically in this type of clock, the time is generated by a model that describes the relation of this clock to another clock (e.g., to a physical clock in a different timescale). The model enables the calculation of the time of the clock from the time of the other clock.”

While time is a complex concept in physics, and generally, even time synchronization is rather ambiguous (what does it mean to be time synchronized in different reference systems?), for some constrained engineering problems in everyday use, we assume Newtonian time perception – time passes by in the same pace in every reference systems and we would like to measure it with different devices. However, even within Newtonian physics, time measurement presents inherent challenges. The recorded time value becomes outdated as soon as it is read or processed since any use of a measured time value

involves processing or computation, which itself takes time. While such issues are beyond the scope of this work, it is important to acknowledge that as accuracy requirements increase, the impact of overlooking these challenges becomes more significant.

Please note that the digital clock in this text does not refer to *digital clock signals*, which are periodic signals toggled between 0 and 1. Digital clocks are concepts closer to clocks used in everyday life – they measure the elapsed time. It is worth mentioning that clock signals can drive clocks; however, they are not the same concept.

To define digital clocks more formally, let us denote time difference from a defined epoch as t , which we will call simply time, but it is important to notice, that this concept is not strictly the same as time in physics. While the epoch can be selected arbitrarily, however, practical applications usually use the so-called UNIX epoch, which is January 1, 1970. A digital clock (or simply clock) is a device, which measures some quantity $C(t)$ depending on time. It is important to emphasize, that from our perspective, clock is a device with dynamics changing with time. C is preferably but not necessarily invertible. Obviously, ideally we would like to have $C(t) = t$. While in history, a couple of dynamic systems were applied to measure time (or more precisely, time difference), e.g. pendulums, water clocks, etc., in practical digital applications nowadays, typically oscillator-based models are applied.

Generally, the oscillator has the output signal as

$$v(t) = V_0 \cos \phi(t) \quad (2.9)$$

where V_0 is the amplitude of the signal, and $\phi(t)$ is the instantaneous phase depending on time t . The function $\phi(t)$ is preferably (and typically) invertible, e.g. it is strictly monotonically increasing. Based on the phase, the instantaneous frequency is defined as

$$f(t) = \frac{1}{2\pi} \frac{\partial \phi(t)}{\partial t}$$

Digital clocks count some defined events of the oscillator at discrete time instants t_k , with $k = 0, 1, 2, \dots$, when the phase holds some specified value, e.g.

$$\phi(t_k) = 2\pi k + c + \Phi_k \quad k \text{ is integer, and } k \geq 0$$

where c is some constant, and Φ_k is some small phase noise (as a random variable) standing for the fact that phase triggers might not be precisely detected. By definition, k is actually the value of the counter and it keeps its value between t_k and t_{k+1} . The actual implementation of this process is out of our interest, e.g. it might be some null crossing detector (i.e. a rising edge of the output of a Schmidt-trigger resulting in the trigger condition $\cos \phi(t) = 0$ and derivative $\sin \phi(t) < 0$). In the next, the c constant will be considered as zero (i.e. $c = 0$), and Φ_k will be ignored, since the synchronization accuracy is couple of orders higher than the noise introduced by mistriggering.

From this, real time t_k can be calculated using the – possibly existing – inverse of $\phi(t)$:

$$t_k = \phi^{-1}(2\pi k) \quad k \text{ is integer and } k \geq 0$$

However, $\phi(t)$ generally is unknown, and we model it as a $\hat{\phi}(t; \boldsymbol{\theta})$ parametric model with parameters $\boldsymbol{\theta}$. We use this parametric model to estimate the time \hat{t} from the k counter as

$$\hat{t}_k = \hat{\phi}^{-1}(2\pi k; \boldsymbol{\theta}) \quad k \text{ is integer and } k \geq 0$$

Based on the estimation of the real-time, we can compute the time error of the clock in theory as

$$\epsilon(t_k) = \hat{t}_k - t_k$$

Please note that the notation above highlights the fact that we consider different errors while estimating time:

Quantization error Thanks to the quantized nature of digital systems, we have no time information between t_k and t_{k+1} for any k .

Triggering error The triggering error represented by Φ_k introduces some unaccounted errors to our time perception, since when we expect the phase to be $2\pi k$ it is actually $2\pi k + \Phi_k$, resulting in estimation errors.

Inversion error The inversion error is actually a modeling error, since we model the unknown clock phase function with a different, parametric function.

In practical applications, the clock frequency and, hence, the clock resolution is a couple of orders higher than the synchronization accuracy (i.e. $t_{k+1} - t_k$ is very small compared to synchronization accuracy), so in this work we neglect the effect of quantization and handle the counter k and the phase as continuous. Also note that the underlying process of digital clocks (i.e., the sinusoidal oscillator) is inherently continuous, supporting the method of handling clocks as continuous processes. Thus, without integer constraint on k , the *counter value* will be considered as

$$k = \frac{\phi(t)}{2\pi} = \int_0^t f(\tau) d\tau \quad (2.10)$$

2.2.1 Ideal oscillator

Ideal oscillator models suppose periodic signal with a known f_0 nominal frequency, i.e.

$$\hat{\phi}(t) = 2\pi f_0 t$$

It is easy to see, that $f(t) = f_0$, $k = f_0 t$ and $\hat{t} = k/f_0$. However, real oscillators cannot be implemented to run with exactly f_0 frequency, so ideal oscillator models are rarely applied in the literature.

2.2.2 Simple skew model

Simple Skew Model (SKM) first appeared in [41] to model imprecise clocks in synchronization applications. The model assumes that the oscillator runs on an unknown

frequency f close to the f_0 nominal frequency, but generally $f \neq f_0$. Also, the model contains an unknown ϕ_0 initial phase component, which is the phase of the oscillator at $t = 0$:

$$\hat{\phi}(t; f, \phi_0) = 2\pi ft + \phi_0 \quad (2.11)$$

From this, $f(t) = f$ and $k = ft + \frac{\phi_0}{2\pi}$. To estimate time based on this model we can use the inverse function, i.e.

$$\hat{t} = \frac{k}{f} - \frac{\phi_0}{2\pi f} \quad (2.12)$$

which of course requires the knowledge of unknown parameters f and ϕ_0 . Sometimes, the error oriented format of this model is used. This uses the difference of the estimated time based on (2.12) and the estimated time using the SKM model, but with nominal frequency $f = f_0$ and $\phi_0 = 0$, which yields

$$\hat{t}_{f_0} - \hat{t} = \frac{k}{f_0} - \frac{k}{f} + \frac{\phi_0}{2\pi f} = \frac{f - f_0}{f_0} t + \frac{\phi_0}{2\pi f_0} = \gamma t + \omega_0 \quad (2.13)$$

where $\gamma = \frac{f - f_0}{f_0}$ is the *skew*, the relative offset of the frequency from the nominal frequency. For the second step in (2.13) we used the fact, that $k = ft + \frac{\phi_0}{2\pi}$. Please note, that this error model is valid only under the assumption of the SKM model (2.11). Simple Skew Model is widely used because of its simplicity and the fact that the dominant error beyond the clock offset in clock models is the frequency skew.

2.2.3 State space probabilistic models

Using probabilistic methods requires digital clocks to be modeled as state space probabilistic models. Since oscillators are inherently continuous processes, modeling can be accomplished using continuous state space models. Since the SKM model fails to grab the effect of intense changing of thermal conditions in the case of typical crystal oscillators, third-order models are commonly applied in the literature, taking clock acceleration into consideration [42, 43].

Denote the clock value at time t as $x(t)$. The state of the clock consists of the clock value and the first two derivatives of the clock value, i.e. $\mathbf{x}(t) = [x(t), \dot{x}(t), \ddot{x}(t)]^T$, where the dot above denotes the time derivative. The general continuous clock model can be represented as a triple integrator of a noise process as

$$\dot{\mathbf{x}}(t) = A^c \mathbf{x}(t) + B^c \mathbf{w}(t) \quad (2.14)$$

where

$$A^c = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \quad B^c = \begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} \quad \mathbf{w}(t) = \begin{bmatrix} w_1(t) \\ w_2(t) \\ w_3(t) \end{bmatrix}$$

and $w_1(t), w_2(t), w_3(t)$ are independent, time-invariant standard Gaussian distributed processes, i.e. for $i = 1, 2, 3$, $w_i(t) \sim \mathcal{N}(0, 1)$ with $\mathbb{E}[w_i(t)w_i^*(t+\tau)] = \delta(\tau)$ (i.e. the delta

function), and $\mathbb{E}[w_i(t_1)w_j^*(t_2)] = 0$ for all $i \neq j$. In the state vector, $x(t)$ is the clock value (i.e. the phase divided by 2π), the $\dot{x}(t)$ is widely known as the clock *instantaneous frequency*, while $\ddot{x}(t)$ is the frequency drift over time. Sometimes, the offset, frequency, and drift errors are modeled, and the states are called offset error, skew, and drift error, respectively.

However, for packet switching protocols, clock synchronization is performed at discrete time points, not continuously. Let's denote the synchronization slot index as k . If the stochastic differential equation in (2.14) is solved starting from some time point t_{k-1} with state \mathbf{x}_{k-1} , then at the time point $t_k > t_{k-1}$, the result is a probability distribution

$$p(\mathbf{x}_k | \mathbf{x}_{k-1}) = \mathcal{N}(A_k \mathbf{x}_k; \Sigma_k) \quad (2.15)$$

To get the A_k and Σ_k matrices, suppose the continuous-time dynamic model as in (2.14). Since the model noise is stationary, we can solve the stochastic differential equation from $t = 0$ to $t = T$ and shift the equations to an arbitrary starting time. Using Laplace transform, and expressing $X(s)$, we get

$$X(s) = (sI - A^c)^{-1} \mathbf{x}(0) + (sI - A^c)^{-1} B^c W(s)$$

where I is the identity matrix, $X(s)$ and $W(s)$ are the Laplace transform of $\mathbf{x}(t)$ and $\mathbf{w}(t)$, respectively. The first member of the sum gives the A_k matrix since it depends on the initial (previous) state, while the second part gives the variance of the noise at time T . Given

$$(sI - A^c)^{-1} = \begin{bmatrix} \frac{1}{s} & \frac{1}{s^2} & \frac{1}{s^3} \\ 0 & \frac{1}{s} & \frac{1}{s^2} \\ 0 & 0 & \frac{1}{s} \end{bmatrix}$$

and noticing that $\frac{1}{s}$, $\frac{1}{s^2}$, $\frac{1}{s^3}$ are the integrals of the unit impulse, and also carrying out the time shift we get the A_k matrix as

$$A_k = \begin{bmatrix} 1 & \Delta_k & \frac{1}{2} \Delta_k^2 \\ 0 & 1 & \Delta_k \\ 0 & 0 & 1 \end{bmatrix} \quad (2.16)$$

where $\Delta_k = t_k - t_{k-1}$.

For the noise covariance matrix Σ_k , notice that the multiplication of $(sI - A^c)^{-1} B^c$ and $W(s)$ is a convolution in the time domain, which can be solved readily. Let $K_s(s) = (sI - A^c)^{-1} B^c$ and $\mathcal{L}^{-1}\{K_s(s)\} = K(t)$, where

$$K_s(s) = \begin{bmatrix} \frac{\sigma_1}{s} & \frac{\sigma_2}{s^2} & \frac{\sigma_3}{s^3} \\ 0 & \frac{\sigma_2}{s} & \frac{\sigma_3}{s^2} \\ 0 & 0 & \frac{\sigma_3}{s} \end{bmatrix} \quad K(t) = \begin{bmatrix} \sigma_1 & \sigma_2 t & \frac{1}{2} \sigma_3 t^2 \\ 0 & \sigma_2 & \sigma_3 t \\ 0 & 0 & \sigma_3 \end{bmatrix}$$

To express the noise covariance matrix, we can express the noise vector as a convolution resulting from $K_s(s)W(s)$:

$$\tilde{\mathbf{w}}(t) = \mathcal{L}^{-1}\{K_s(s)W(s)\} = \int_0^T K(T - \tau) \mathbf{w}(\tau) d\tau$$

The covariance matrix of $\tilde{\mathbf{w}}$ is the expression

$$\begin{aligned}\mathbb{E}[\tilde{\mathbf{w}}(t)\tilde{\mathbf{w}}^*(t)] &= \mathbb{E}\left[\int_0^T K(T-\tau_1)\mathbf{w}(\tau_1)d\tau_1 \int_0^T \mathbf{w}^*(\tau_2)K^*(T-\tau_2)d\tau_2\right] = \\ &= \int_0^T \int_0^T K(T-\tau_1)\mathbb{E}[\mathbf{w}(\tau_1)\mathbf{w}^*(\tau_2)]K^*(T-\tau_2)d\tau_1d\tau_2 = \\ &= \int_0^T K(T-\tau_2)K^*(T-\tau_2)d\tau_2\end{aligned}$$

because $\mathbb{E}\{\mathbf{w}(\tau_1)\mathbf{w}^*(\tau_2)\} = I\delta(\tau_2 - \tau_1)$. Performing the integration yields

$$\Sigma_k = \mathbb{E}[\tilde{\mathbf{w}}(t)\tilde{\mathbf{w}}^*(t)] = \begin{bmatrix} \sigma_1^2\Delta_k + \sigma_2^2\frac{\Delta_k^3}{3} + \sigma_3^2\frac{\Delta_k^5}{20} & \sigma_2^2\frac{\Delta_k^2}{2} + \sigma_3^2\frac{\Delta_k^4}{8} & \sigma_3^2\frac{\Delta_k^3}{6} \\ \sigma_2^2\frac{\Delta_k^2}{2} + \sigma_3^2\frac{\Delta_k^4}{8} & \sigma_2^2\Delta_k + \sigma_3^2\frac{\Delta_k^3}{3} & \sigma_3^2\frac{\Delta_k^2}{2} \\ \sigma_3^2\frac{\Delta_k^3}{6} & \sigma_3^2\frac{\Delta_k^2}{2} & \sigma_3^2\Delta_k \end{bmatrix} \quad (2.17)$$

From the covariance matrix, it is readily visible that the uncertainty of the clock value is increasing monotonically with time. The previously presented Simple Skew Model can also be represented as a state space probabilistic model using the very same steps for a second-order linear differential equation.

2.2.4 Measuring frequency stability

The most important clock deviation factor is the instantaneous frequency difference between two physical clocks. While the frequency of real oscillators can be tuned, it is never possible to reach identical frequencies for two clocks without some feedback-controlled mechanism since the frequency of the oscillator starts wandering.

To measure the frequency stability of different oscillators, typically $\phi(t)$ phase and $\bar{f}(t) = \frac{f(t)-f_0}{f_0}$ fractional frequency measurements are used [44]. In real life, it is common to sample these quantities with τ_0 period, resulting in measurements ϕ_i and \bar{f}_i , where i refers to the data point at time $i \cdot \tau_0$, i.e. $\phi_i = \phi(i\tau_0)$ and $\bar{f}_i = \bar{f}(i\tau_0)$.

Most of the time, the simplest maximum allowable frequency deviation from the nominal frequency is enough to describe the oscillator stability. The *tolerance* is specified for a given temperature (usually 25 °C), and defined as

$$\max |\bar{f}(t)|$$

Tolerance is commonly measured in ppm (parts per million). For typical values of frequency tolerance of practical oscillators, see Figure 2.2. Commercial products rarely contain more stable oscillators than [Temperature-compensated crystal oscillator \(TCXO\)](#), however, some higher line products may contain [Oven-controlled crystal oscillator \(OCXO\)](#), but atomic solutions are rather expensive. Thus, only very specific devices

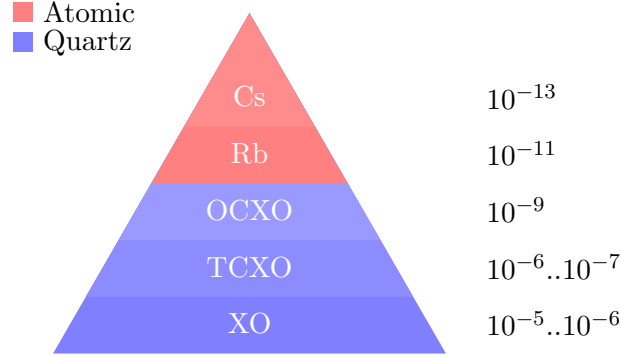


Figure 2.2: Relative frequency tolerance of common oscillator types. Tolerance reaches from hundreds of ppm to several ppb in case of crystal oscillators, while atomic oscillators are in the order of ppt [45]. (Cs – Cesium, Rb – Rubidium, OC – Oven Controlled, TC – Temperature Compensated, XO – Crystal Oscillator)

contain – possibly chip-scale – atomic clocks. Also note that lower tolerance typically comes with higher energy consumption, e.g. OCXOs need high current to reach the controlled temperature inside the packaging.

One of the detailed descriptions of frequency stability is the *variance*, namely

$$s^2 = \frac{1}{M-1} \sum_{i=1}^M (\bar{f}_i - \frac{1}{M} \sum_{i=1}^M \bar{f}_i)^2$$

where the summation is done for M data points measured at i discrete time points and \bar{f}_i is the fractional frequency at the i time point. Standard variance, however, is not convergent for different types of errors (e.g. FM noise), so more commonly, the *Allan variance* (or its square root, the *Allan deviation*) is used for measuring frequency stability, i.e.

$$\sigma_{yo}^2(\tau_0) = \frac{1}{2(M-1)} \sum_{i=1}^{M-1} (\bar{f}_{i+1} - \bar{f}_i)^2 = \frac{1}{2(M-2)\tau_0^2} \sum_{n=1}^{M-2} (\phi_{n+2} - 2\phi_{n+1} + \phi_n)^2$$

In practice, *Overlapping or Overlapped Allan variance* is more widely used, which fits better for sampled measurements with fixed time intervals and has better confidence. The Overlapped Allan variance is defined as

$$\sigma_y^2(m\tau_0) = \frac{1}{2(m\tau_0)^2(M-2m)} \sum_{n=1}^{M-2m} (\phi_{n+2m} - 2\phi_{n+m} + \phi_n)^2$$

which is defined at multiples of τ_0 , i.e. at $m\tau_0$.

Figure 2.3 shows an Allan variance plot at different values of τ_0 . It is clearly visible that the Allan deviation has a minimum, before which the instantaneous frequency noise

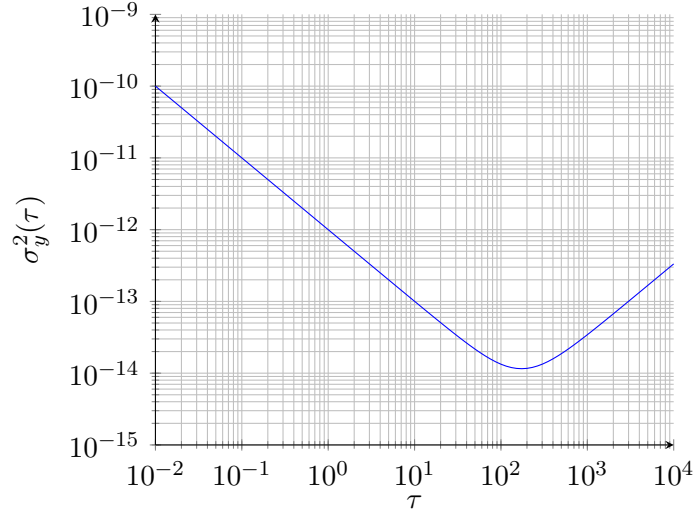


Figure 2.3: Allan variance of a typical crystal oscillator ($\sigma_1 = 10^{-6}, \sigma_2 = 10^{-8}$).

dominates, but at high values of τ_0 , the aging or wandering of the frequency takes over the dominance.

Allan-variance can be strongly related to probabilistic state space models, as presented in (2.14), for which the Allan-variance equals (see [42])

$$\sigma_y^2(t_k, \tau) = \left(\frac{\sigma_1^2}{\tau} + \frac{\sigma_2^2 \tau}{3} + \frac{\sigma_3^2 \tau^3}{20} \right) + \sigma_3^2 \left(\frac{\tau^3}{3} + \frac{\tau^2 t_k}{2} \right) \quad (2.18)$$

assuming that $\ddot{x}(0) = 0$. Please note that the Allan-variance in (2.18) is not time-invariant, however, if $\sigma_3 = 0$, i.e. we ignore the acceleration of the oscillator, resulting in the well-known expression for the time-invariant Allan-variance

$$\sigma_y^2(\tau) = \frac{\sigma_1^2}{\tau} + \frac{\sigma_2^2 \tau}{3} \quad (2.19)$$

In the literature, a couple of other measures for frequency stability can be found, e.g. the Modified Allan deviation, the Hadamard Variance, Th  o1 variance, and so on. To get more insight in the frequency stability measures, the reader is referred to [44].

2.3 Clock synchronization

Clock synchronization has a few definitions, providing the description of the more-or-less same process. The IEEE 1588-2019 standard [11] describes two clock to be synchronized as this:

„Absent relativistic effects, two clocks are *synchronized* to a specified uncertainty if they have the same epoch and their measurements of the time of the

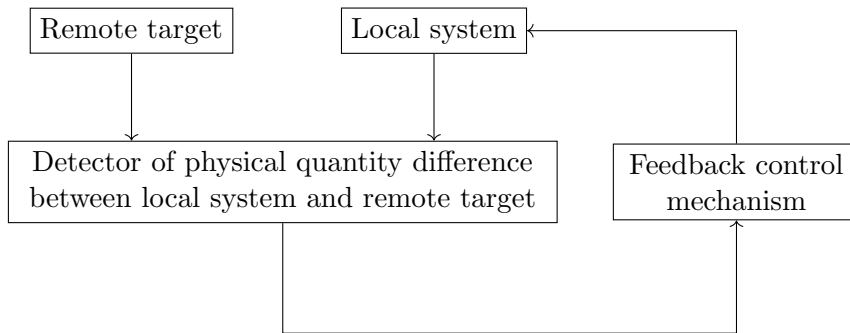


Figure 2.4: The feedback process of synchronization in general (source: [46]).

same single event occurring at an arbitrary instant differ by no more than that uncertainty.”

In fact, we say that two digital clocks are synchronized if, in the case of a real event, the two clocks show the very same value. However, the definition has two more constraints: relativistic effects shall be neglected, and synchronization shall not be completely perfect – in the case of predefined limits, two clocks can be synchronized to a given uncertainty.

Sometimes, only the speed (i.e. frequency) of the clocks shall be synchronized, literally relieving the constraint of the „same epoch”. We call frequency synchronization as *synthonization*, defined in the IEEE 1588-2019 standard:

„Absent relativistic effects, two clocks are *synthonized* to a specified uncertainty if the duration of the second is the same on both, which means the time as measured by each advance at the same rate within the specified uncertainty. The two clocks might or might not share the same epoch.”

Once again, it is useful to refer to Section 2.2, where certain thoughts on time measurement were discussed. This work focuses on the constrained engineering problem of clock synchronization within the framework of Newtonian time perception. In this context, time progresses uniformly across all reference frames, and the goal is for all clocks to display the same value at a given moment. Put simply, from a physical perspective, clock synchronization is simply the process of aligning the dynamics of two remote devices using digital communication. The concept of a clock itself only emerges when considering dynamics *in time*, which is assumed to be partially known but well-defined.

Clock synchronization typically assumes that there is a reference clock to which the clock at hand shall be synchronized by various means. The reference clock can be considered to follow time with a small uncertainty. However, it can eventually be ignored in the clock synchronization algorithm, the only thing we are interested in for the two devices have the same dynamics. The synchronization process is inherently a feedback process, as seen in Figure 2.4. The local clock – which is to be synchronized with the remote clock – measures the discrepancy of the local clock to the remote clock and takes some action to decrease the discrepancy. However, changing any parameters of the local clock results in changes in the discrepancy, which should be periodically managed.

Time synchronization over packet-switching networks has some specialties. First, the synchronization happens in discrete time intervals or at discrete events – compared to, e.g., a [Phased-locked loop \(PLL\)](#) based continuous circuit. Second, a clear distinction shall be made between the algorithm providing the clock synchronization and the communication infrastructure supporting the utility tasks of the clock synchronization. For example, while the IEEE 1588-2019 standard provides all the standard tools to identify clocks and communicate parameters of the clocks and the standard provides even various message exchanges to enable the measurement of the propagation delay, it does not define the compulsory method of tuning the clock to be synchronized.

This work concentrates mainly on the algorithmic part, however, no synchronization algorithm can skip the ins and outs of the medium providing the measurements for clock synchronization. For this purpose, the next subsection provides an overview of the well-known IEEE 1588-2019 protocol, which serves as a baseline for accurate clock synchronization and integrates prior knowledge from various time synchronization platforms and solutions.

2.3.1 IEEE 1588 standard

The *IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems* (IEEE 1588), sometimes called *Precision Time Protocol*, is a protocol that provides precise synchronization of clocks in packet-based networked systems. The synchronization of clocks can be achieved in heterogeneous systems that include clocks with different inherent precision, resolution, and stability. The standard had a couple of versions, as

IEEE 1588-2002 The first version of the standard, known as [PTP](#) version 1. [24]

IEEE 1588-2008 The new standard, known as [PTP](#) version 2 introduced a couple of new concepts, like transparent nodes in the network. Not backward compatible with [PTP](#) version 1. [47]

IEEE 1588-2019 The latest standard, also known as [PTP](#) version 2.1 is backward compatible with [PTP](#) version 2, and introduces isolated profiles, multiple masters and security methods. [11]

The standard contains the description of the data types, the transport, the protocol for different clocks, the definition of the PTP entities, and the required data sets for various clocks. The PTP protocol can use various transport protocols to pass PTP messages between PTP entities; the standard defines, e.g., the PTP over [UDP](#) and the PTP over IEEE 802.3 (Ethernet) methods.

The standard defines different types of clocks, which build an acyclic hierarchy in a so-called PTP domain. The different clock types can be:

Ordinary clock Ordinary clocks have a single PTP port in a domain and maintains the timescale used in the domain. It may serve as a source of time (i.e. master clock), or may synchronize to another clock (being a slave clock).

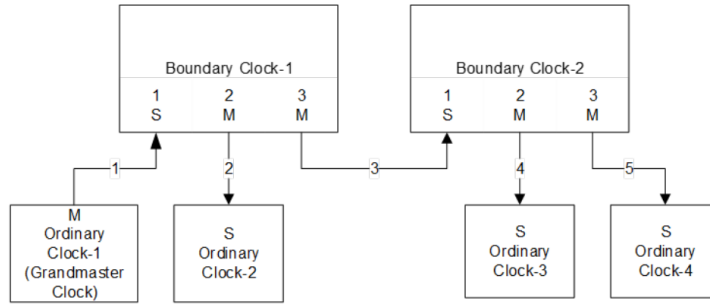


Figure 2.5: Simple master-slave PTP Instance hierarchy (source: [11]).

Boundary clock Boundary clocks have multiple PTP ports in a domain and maintain the timescale used in the domain. One of the ports of the boundary clock can act as a slave, and the other ports act as masters synchronized to the only slave port.

Transparent clock Transparent clocks can measure the time residence, e.g. the difference of the egress and ingress time.

In fact, transparent clocks can be End-to-end or Peer-to-peer, where the latter can measure not only the residence of the message in the device, but using a peer-to-peer mechanism, it can also provide corrections to the propagation delay of the link connected to the PTP ports.

A typical PTP hierarchy can be seen in Figure 2.5, which shows the relation of ordinary and boundary clocks. In every PTP domain, the root of the synchronization hierarchy, a grand master clock, can be found. The organization of the master-slave hierarchy is dynamic, and the selection of the master clock for each port is done by the Best Master Clock selection algorithm, using special *Announce* messages to transfer information about the datasets of the clocks.

The actual measurements for the synchronization consist of inferring information from the actual clock value at the master and the calculation of the path delay between the master and the slave. From that, the actual time at the slave can be acquired. Figure 2.6 shows the end-to-end measurement message exchange defined in the PTPv2 standard. The standard uses two essential messages to achieve end-to-end synchronization, a *Sync* message and a *Delay_Req* message. The *Sync* message is sent by the master, and the master notes the transmission (TX) timestamp ($t^{(1)}$) of the message. The slave device receives the message and saves the receive (RX) timestamp ($t^{(2)}$) of the message. In the case of one-step implementation, the TX timestamp is embedded in the *Sync* message, while for two-step implementation, an auxiliary message (*Follow_Up*) is used to send the TX timestamp to the slave. While the clock drift (the frequency difference) can be derived from the *Sync* message, an additional *Delay_Req* message is required to estimate the clock offset. The *Delay_Req* is sent by the slave at $t^{(3)}$ and received by the master at $t^{(4)}$, which is sent back to the slave in an auxiliary message *Delay_Resp*. From the recorded timestamps, the clock offset and clock drift can be calculated. To calculate

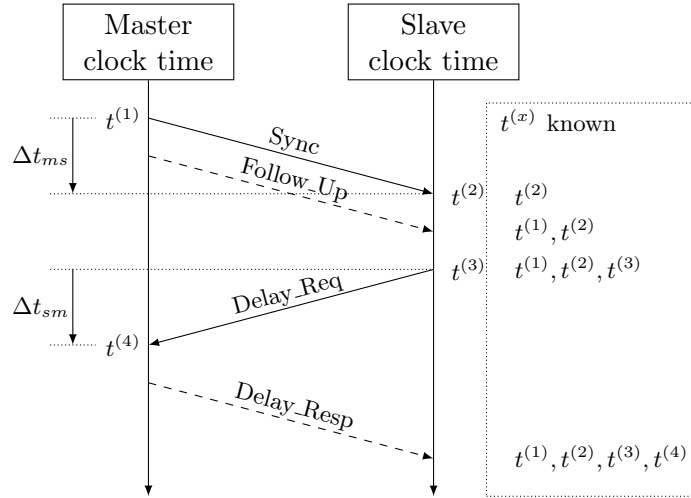


Figure 2.6: End-to-end PTP path delay measurement message exchange (source: [11]).

the offset value from the PTP master, the master-to-slave and the slave-to-master offset need to be defined:

$$d_{m2s} = t^{(2)} - t^{(1)}$$

$$d_{s2m} = t^{(4)} - t^{(3)}$$

From this, the offset from the master can be calculated:

$$\Delta t_{\text{offset}} = d_{m2s} - d_{\text{prop}} = d_{m2s} - \frac{d_{m2s} + d_{s2m}}{2} \quad (2.20)$$

where d_{prop} is the propagation delay.

Achieving nano- or even microsecond accuracy is beyond the capability of software-only solutions. Therefore, hardware-assisted methods are preferred, leveraging the inherent precision of modern CPU architectures. Hardware-aided methods make it possible to precisely record the TX and RX timestamps for packets transmitted to or received from the network; however, the synchronization algorithm can be implemented in software. Moreover, there is the White Rabbit project – included in IEEE 1588-2019 – where sub-nanosecond accuracy can also be achieved on a specialized network [11]. White Rabbit is a project aimed at creating an Ethernet-based network that provides low-latency, deterministic packet delivery, and transparent, high-accuracy timing distribution across the entire network. The White Rabbit Network is based on Ethernet (IEEE 802.3), [Synchronous Ethernet \(SyncE\)](#), and PTP standards. The measured PTP performance demonstrates sub-nanosecond accuracy over a 5km fiber optic link with a precision below 10 ps [3].

Accurate clock synchronization and PTP's main application area is in the industrial domain, especially in [Time-Sensitive Networking \(TSN\)](#). TSN is a set of standards under development by the Time-Sensitive Networking task group of the IEEE 802.1 working

group [48]. Time-Sensitive Networking Task Group specifies TSN functionalities as a set of standards that provide deterministic services through IEEE 802 networks to enable bounded low packet loss, guaranteed packet delivery, bounded low latency, and low packet delay variation. TSN targets a variety of particular aspects, including timing and synchronization aspects. Synchronization is an essential building block to meet these requirements, and IEEE 802.1AS-2020 standard [27] defines a PTP profile to use in TSN networks.

For more information about the protocol, e.g. peer-to-peer measurements, profile management, the exact message formats, and transparent clock correction mechanism, please see the IEEE 1588-2019 standard [11].

2.3.2 Controllable and virtual clocks

Clock synchronization is typically performed by adjusting the frequency of a physical clock to skew it to the reference time. This results in a smooth transition that is required in most use cases. However, there are systems where a physically adjustable clock is not present, e.g., in some embedded systems or even in personal computers without [PTP](#) clocks. In this case, a so-called *virtual clock* can be employed, which transforms the local, non-adjustable clock to a reference clock.

Virtual clock

In the case when no physically adjustable clock is present, based on the local clock, the clock models presented in Section 2.2 can be employed. The estimation of the parameters can be done using various methods, like Kalman filtering, regression, etc. Every time the system wants to know the exact time, the clock models can be used by reading the local clock value, and the elapsed local time can be substituted into the model to get the value or the probability distribution of the actual time inferred from the measurements from the reference clock.

Clock control

If a system has a physically adjustable clock (e.g. frequency-adjustable clock), it is required that the clock synchronization algorithm adjust the clock so that it would reflect the time measured by the reference clock. These algorithms are commonly called *clock servos*.

Typical clock servo solutions widely use [Proportional Integral \(PI\)](#) controllers as shown in Figure 2.7a. The clock is modeled as an ideal integrator, and the reference signal for the controller is the packet delay corrected time of the reference clock. Notice that the synchronization is automatically performed by the control loop itself. However, there are two main pitfalls of a bare PI controller:

1. In classic control loops, the reference signal is deterministic; however, in the case of clock synchronization, the reference signal is a stochastic process.

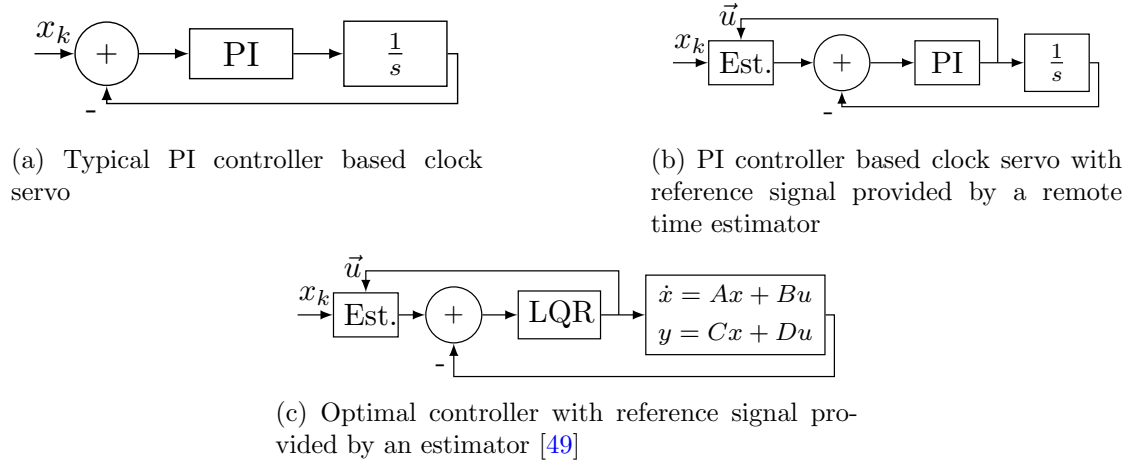


Figure 2.7: Different clock servo architectures to control slave clock with adjustable frequency. Statistical estimators provide approximate estimation to the remote time based on the local clock.

2. The reference signal is dependent on the control signal since the packet delay estimation contains measurements from the clock itself.

To solve these issues, the reference signal shall be estimated by a statistical estimator, e.g., Kalman filtering. In dynamic systems, the estimator dynamic model needs to be modified to take the control signal into account, modifying the dynamic model (see (2.15)) as

$$p(\mathbf{x}_{k+1}|\mathbf{x}_k) = \mathcal{N}(A_k\mathbf{x}_k + B_k\mathbf{u}_k; \Sigma_k) \quad (2.21)$$

with the A_k matrix as in (2.16), $\mathbf{u}_k = [\theta_k \ \gamma_k \ \alpha_k]^T$ being the control signal, and

$$B_k = \begin{bmatrix} -1 & -\Delta_k & -\frac{1}{2}\Delta_k^2 \\ 0 & -1 & -\Delta_k \\ 0 & 0 & -1 \end{bmatrix} \quad (2.22)$$

The θ_k is the phase control, γ_k is the skew control, α_k is the skew drift control signal. In real implementations, typically $\alpha_k = \theta_k = 0$, and γ_k controls the frequency of the clock. The architecture utilizing dynamic estimators to provide the reference signal can be seen in Figure 2.7b. The method of creating the reference signal is sometimes called *trajectory generation* in control theory.

Optimal control can be provided in a least-square sense by using [Linear Quadratic Regulators \(LQR\)](#). LQR requires a state space model with linear state feedback, which can be found by solving the *discrete time algebraic Riccati equation*. The optimal control loop assumes the modeling of the clock as a state-space model. The special architecture of the control loop can be found in Figure 2.7c and is described in detail in [49]. For more information regarding the method of designing an optimal clock controller, please see [50, 51].

2.3.3 Clock state estimation algorithms

Clock synchronization encounters two primary sources of uncertainty: the clock phase itself has a random behavior [42], and the computed offset based on PTP measurements is strongly dependent on the packet delay variation. While propagation delay arises in any systems where signals propagate, in packet-switched systems, we distinguish the so-called **packet delay variation (PDV)** from propagation delay. The packet delay variation is inherently random and caused by several factors: e.g., queuing delays, processing delays, software processing delays (i.e., scheduling, interrupt latency, etc.), or, as in wireless systems, multipath propagation, resulting in a diverse measurement noise [52, 53]. In many applications, such as wireless localization, the analysis of propagation delay is crucial, however, in wired systems, like PTP, the focus is on the packet delay variation (since propagation delay is typically constant). To overcome the issue of packet delay variation, several offset estimation algorithms have been previously proposed, assuming various clock and measurement models. An exhaustive summary of proposed offset estimators can be found in [54] and [55].

Besides the proposed default estimator in the IEEE 1588-2019 standard, more sophisticated methods include window-based processing techniques, which select the best possible measurements to infer the actual time. These methods are commonly called „packet selection” methods, and they use different statistics to sample a window for potential measurements. One of its prominent examples is the Minimax Optimum Estimator [56], which applies various statistics (e.g., minimum, maximum, median) to the master-to-slave and the slave-to-master delays. However, the performance of window-based methods strongly relies on the window length. Assuming linear clock models, linear least-squares solutions propose to estimate the model parameters minimizing the least-square error using closed-form solution or linear programming [57]. **PTP-LP** method also provides a heuristic way of estimating clock parameters using linear regression instead of solving the linear program itself [58]. However, linear least square solutions tend to fail in situations when the measurement error is not Gaussian or the clock behaves non-linear. Anantha et al. [59] showed that the random queuing delay in the packet delay variation can be approximated by a mixture of Gaussian distributions. However, the method requires running a generalized expectation-maximization algorithm on a decent state space, and the method is evaluated only by simulations. Exel et al. [60] handles the problem of unknown clock parameters using an optimized servo parametrization, but it still requires the exact estimation of the packet delay variation.

To deal with random noise, Kalman filtering was soon applied to clock synchronization, proven to be more robust than the classical IEEE 1588 standard-based methods [61]. The basic work for Kalman-filtering in clock synchronization is based on a second-order Kalman-filter and the original IEEE 1588 measurement model [62, 63]. Kalman filters require the exact estimate of the measurement noise variance to be optimal in the least-square sense and to overcome this issue, adaptive and robust methods were proposed. Adaptivity and robustness have different meanings in the literature: adaptive filtering estimates the process and measurement noise covariance, while robust filtering handles outlier measurements. Robust filtering is widely applied in Kalman filters for PTP

time synchronization using innovation-based outlier detection [61, 64] or timestamp selection [65]; however, robust methods work well only if the measurement noise covariance is still known a priori.

2.3.4 Precision and accuracy

Accuracy and *precision* occupy an important place in the literature of clock synchronization, so it is worth taking some notes about these concepts. While the IEEE 1588-2008 standard defines accuracy as „The mean of the time or frequency error between the clock under test and a perfect reference clock, over an ensemble of measurements”, and precision is „a measure of the deviation of the error from the mean”, there are a couple of misconceptions about these notions [66], and to circumvent this, also *trueness* is introduced. It is interesting to emphasize that the IEEE 1588 protocol called the Precision Time Protocol bears the name precision, while the protocol’s main goal is to provide means to enable accurate (and not precise on the first hand) time synchronization. However, the reader shall have a clear distinction between the two concepts throughout this text.

To ease the burden of these concepts, this work uses *bias* and *variance*, which have clear definitions in statistics. Let $\hat{\theta}(X)$ be an estimator of the unknown value θ based on the random variable X . The bias of the estimator is defined as $\mathbb{E}[\hat{\theta}(X)] - \theta$, while the variance of the estimator is defined as $\mathbb{E}[(\hat{\theta}(X) - \mathbb{E}[\hat{\theta}(X)])^2]$, where $\mathbb{E}[\cdot]$ is the expectation operator. For clarity, if we want to emphasize the difference between the mean error and the error deviation, the concepts of bias and variance (or its square root, the *deviation*) are used instead of accuracy and precision.

2.4 Ultra wide band communication

Ultra-Wideband (UWB) communication is a short-range communication form using very low energy level but very high bandwidth, typically greater than 500 MHz [67]. The emission limit for UWB transmitters is -41.3 dBm/MHz, which makes it possible to be under the noise floor defined by the Federal Communications Commission (FCC), thus not interfering with other transmitters around (e.g. WiFi, cell phones, GPS) as Figure 2.8 shows.

With a huge bandwidth of 500 MHz (or 0.2 fractional bandwidth) or greater, the UWB technology obviously promises robustness against narrowband interference, small-scale fading, multi-path immunity, and strong obstacle penetration during communication and distance measurements. Although UWB suffers some applicability constraints, such as high hardware cost, short-range communication, and relatively high energy consumption [68], its advantages in industrial and consumer systems for localization are undeniable. UWB is a potential candidate for implementing asset management, assembly control, or smart manufacturing for a variety of different industries [69–71], consumer products [72], as well as short and medium-range communication [73, 74].

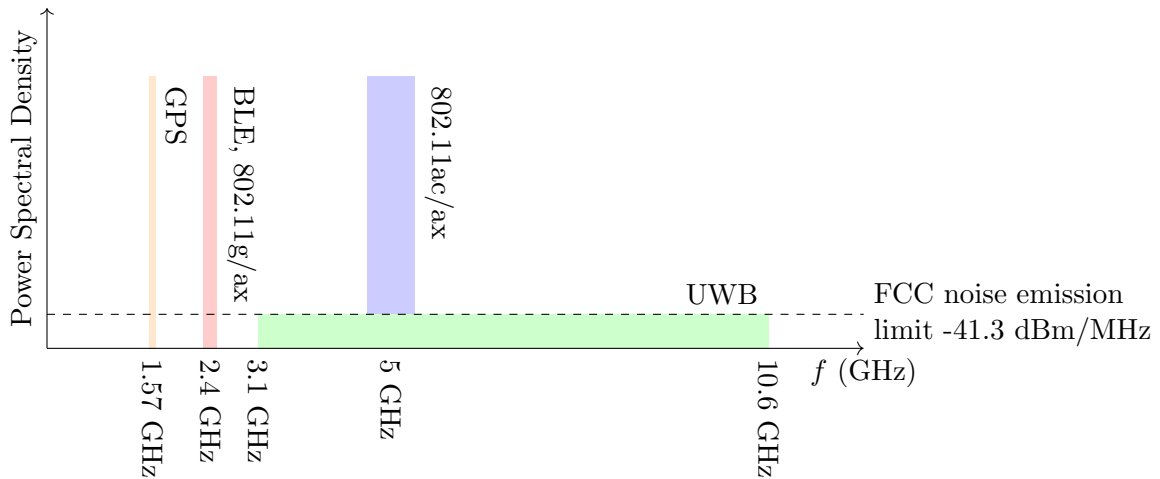


Figure 2.8: The ultra wide band communication makes transmission possible under the noise floor (green area), thus not interfering with other technologies. Note: magnitudes are for illustrational purposes.

The UWB bands can be found in the sub-gigahertz band (249.6 - 749.6 MHz), the low band (3.1-4.8 GHz), and the high band (6.0-10.6 GHz). Bandwidth varies from 499.2 MHz to 1354.97 MHz. The UWB is a physical layer, which is standardized as a physical layer for the IEEE 802.15.4 standard, which defines the [high-rate pulse repetition frequency UWB \(HRP UWB\)](#) and the [low-rate pulse UWB \(LRP UWB\)](#) PHY layers [14]. The bit rate per second can vary from hundreds of kbps to several Mbps. Higher layer functions (e.g. medium access layer) are defined by the standard.

The most interesting function of the UWB physical layer is the capability of very precise timestamping of UWB messages, providing accuracy in the order of 100 picoseconds. This function is enabled by the very short pulses that the UWB applies to transfer information through the radio channel. The standard discusses the ranging capability as the main use case for accurate timestamping, which is implemented by a high-resolution counter with the [least significant bit \(LSB\)](#) as long as $1/(128 \cdot 499.2 \cdot 10^6 \text{Hz}) \approx 15 \text{ ps}$ (see [14]). However, the clock drifts in the devices pose a great challenge against the ranging accuracy despite the fine resolution of the clock counters.

Figure 2.9a shows the default two-way ranging method for estimating the range between two devices. The measured time-of-flight can be expressed as

$$T_f = \frac{1}{2}(T_R^{(1)} - T_D^{(2)}) \quad (2.23)$$

where $T_R^{(1)}$ denotes the round trip time at device 1 and $T_D^{(2)}$ denotes the delay time at device 2, T_f is the [time-of-flight \(ToF\)](#). From T_f , the range s can be computed by multiplying with the speed of light, i.e. $s = c \cdot T_f$. While the measurement accuracy of the timestamping is also a fascinating topic (see Chapter 5), the frequency drift can also result in a huge-ranging error, as shown below.

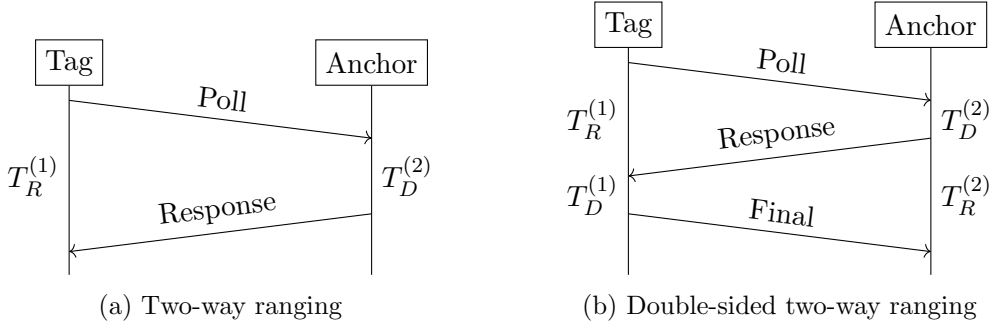


Figure 2.9: Message exchange of different ranging techniques in UWB systems.

Modeling clock drift as a simple relative error (SKM model), the round-trip time and the delay can be modeled as

$$\hat{T}_R^{(1)} = (1 + e_1)T_R^{(1)} = k_1T_R^{(1)} \quad (2.24)$$

$$\hat{T}_D^{(2)} = (1 + e_2)T_D^{(2)} = k_2T_D^{(2)} \quad (2.25)$$

which yields

$$\hat{T}_f = \frac{1}{2}(\hat{T}_R^{(1)} - \hat{T}_D^{(2)}) \quad (2.26)$$

from which we can calculate the time-of-flight error as

$$\hat{T}_f - T_f = e_1T_f + \frac{T_D^{(2)}}{2}(e_1 - e_2) \quad (2.27)$$

Supposing $T_D^{(2)} = 1$ ms, and typical ± 20 ppm oscillators, the worst case error for $e_1 - e_2$ is ± 40 ppm, which results in a time-of-flight error $|\hat{T}_f - T_f| > 20$ ns, or almost 7 meters. However, using double-sided two-way ranging (see Figure 2.9b), and an alternative calculation of the time-of-flight, the frequency drift error can be compensated [75]:

$$T_f = \frac{T_R^{(1)}T_R^{(2)} - T_D^{(1)}T_D^{(2)}}{2(T_R^{(1)} + T_D^{(1)})} \quad (2.28)$$

where $T_R^{(1)}$ and $T_R^{(2)}$ are the round-trip times, and $T_D^{(1)}$ and $T_D^{(2)}$ are the delay times at devices 1 and 2, respectively. Here, the time-of-flight estimation error is

$$\hat{T}_f - T_f = e_1T_f \quad (2.29)$$

which means that the ranging error is proportional to the frequency drift error.

However, it is important to note that this method works only in the case of direct ranging. More energy-conservative localization solutions are also applicable in UWB systems (i.e., time difference of arrival techniques), which require estimating the frequency drift in order to achieve centimeter-accurate locations.

3

Efficient ultra wide band measurement system design

Typical ranging-based **UWB** localization architecture consists of anchors that are fixed and tags that have unknown locations. Anchors and tags communicate with each other using short UWB pulses. UWB supports any localization method that is based on the accurate detection of signal arrival time, such as **time of arrival (ToA)** and **time difference of arrival (TDoA)**. While implementation of TDoA is more efficient regarding radio resources, ToA based implementations outperform TDoA in terms of accuracy and robustness, thus are used as an industrial standard for localization [76]. The most robust and accurate implementation of ToA is the classical **double sided two-way ranging (DS-TWR)** method [14] - called **TWR** in the followings -, which can determine distance between anchor-tag pairs [77].

However, any localization system over the UWB technology requires some time synchronization solution, explicitly (like ToA systems) or implicitly (e.g. ranging). The synchronization and ranging capabilities of the system strongly depend on the frequency of the clock state measurements, namely the amount of message exchanges between the devices involved in the localization. The classical application of TWR wastes radio resources since localization requires measuring at least three distances for two-dimensional positioning, and measuring one distance requires at least three signals between each tag-anchor pair. The disadvantage of this approach is the time separation of tag ranging, which limits the number of tracked tags or, by increasing the number of tags, significantly decreases the ranging frequency.

In the last decade, researchers made great efforts to increase the accuracy and robustness of UWB systems. Recent surveys and reviews on UWB technology focus on ranging accuracy [21], especially in **non-line of sight (NLOS)** environments [19], robust localization mechanisms introducing filters [78] or deep-learning concepts [79]. In the literature,

efficient utilization of radio resources mainly focuses on minimizing the ranging time by reducing the size of messages sent by participating tags [80], but resource-efficient system design and scalability issues were not in the research focus.

This chapter introduces a messaging framework that optimizes the usage of resources. The advantage is outstanding when it is compared to a classical TWR-ranging protocol efficiency. However, this messaging design introduces additional ranging delay while increasing the scalability of ToA systems. To justify the virtue of the proposed framework, ranging results are compared with classical distance measurements. The related experimental performance evaluation proves that the introduction of higher ranging time does not affect the ranging accuracy; however, it drastically increases scalability.

3.1 Increasing the effective data rate for enhancing measurement frequency

To introduce the proposed solution for incrementing the effective data rate, we shall shortly discuss the anatomy of a UWB packet [14]. Typical UWB packet consists of a **Synchronization header (SHR)**, **PHY header (PHR)**, and the remaining user data (**PHY Service Data Unit (PSDU)**). The SHR part is always transmitted with a common rate independent of the data rate (Table 3.1). In robust systems, using 1024 symbols long preamble with a typical 20 bytes of data, the preamble is $(1024 \cdot 993.59 \text{ ns})$ $1017 \mu\text{s}$ long while data is transmitted in $(20 \cdot 8 \cdot 1025.64 \text{ ns})$ $164 \mu\text{s}$, given the use of 850 kbps data rate. The example shows that the preamble occupies the channel most of the time, which even happens to be true when shorter preambles are used. The degradation of the effective data rate is worse in the case of higher data rates.

The exact rates can be inferred from the DW1000 datasheet [81]. For a 850 kbps and 16 MHz **pulse repetition frequency (PRF)** the transmission time can be calculated approximately as

$$\begin{aligned} n_{\text{reed}} &= 48 * (8l + 329)/330 \\ \tau(l) &= 993.59(n_{\text{pr}} + 8) + 1025.64 \cdot 21 + 1025.64(8l + n_{\text{reed}}) = \\ &= \alpha l + \gamma n_{\text{pr}} + \text{const.} \end{aligned} \quad (3.1)$$

where n_{pr} is the preamble length in bits and l is the length of the payload in bytes, α and γ are constants. The main thing to note here is that the time required to transmit a message depends linearly on the l length and it also requires some additional constant time.

A typical ranging-based localization system is composed of tags and anchors. Anchors are fixed installed devices with known locations, while tags are moving devices to be localized by the localization system. Generally, a ranging-based localization system uses four ranging measurements per tag to provide robust localization. To save some air time, a common practice is to transmit a grouped poll message and a grouped final message (e.g. method used by Decawave [82]). Figure 3.4 shows an example of this message order in a localization architecture with four anchors while two tags are localized. It is

Table 3.1: The symbol duration in the IEEE 802.15.4-2011 standard for the UWB PHY at PRF 16 MHz

Data rate (Mbps)	SHR (ns)	PHR (ns)	Data (ns)
0.11	993.59	8205.13	8205.13
0.85	993.59	1025.64	1025.64
6.81	993.59	1025.64	128.21

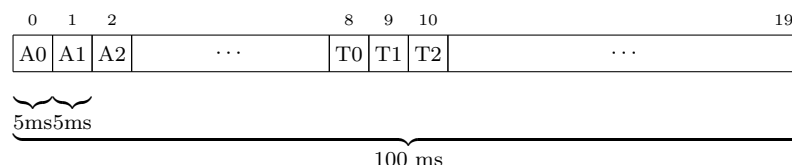


Figure 3.1: The proposed superframe structure. The superframe can be partitioned into an anchor part and a tag part. In the superframe, anchors send their messages in sequential order, and then the tags send their messages in sequential order. In the figure, 'A' stands for anchor, and 'T' stands for tag. The example shows 8 anchors and 12 tags, resulting in a 100 ms long superframe with 5 ms timeslots.

worth noticing that this method requires three dedicated messages for each and every tag to implement robust localization, which is the main reason why this method is rather wasteful.

To overcome this waste, the main idea is to send more data in one UWB packet PSDU; thus, the effective data rate can be increased. To achieve this a framing technique is constructed which is capable of sending more data per packet while maintaining the ability to perform TWR, as well.

3.1.1 Proposed solution

To analyze a robust implementation, the calculations assume an 850 kbps data rate and 1024 symbol-long preamble, but the results are applicable to other configurations as well. The proposed superframe structure can be seen in Figure 3.1. The structure of the superframe consists of an anchor part and a tag part. During the anchor part, the anchors send their messages, while in the tag phase, tags send their messages. The transmissions happen in sequential order, which, of course, assumes that the devices own some preconfigured identifiers. The messages contain a frame control part (for identifying the message type), the sender identifier and a monotonically increasing transaction id.

The slots in the superframe are 5 ms wide, which provides a decent guard time between slots. For example, for 8 anchors and 12 tags, the whole superframe is 100 ms long, so in one second, 10 of this superframe can be transmitted. The format of the anchor and tag packets can be found in Figure 3.2. In the proposed system, there are only broadcast messages identified by a transaction identifier that includes the sequence number of the actual superframe. Each message contains the transmission timestamp

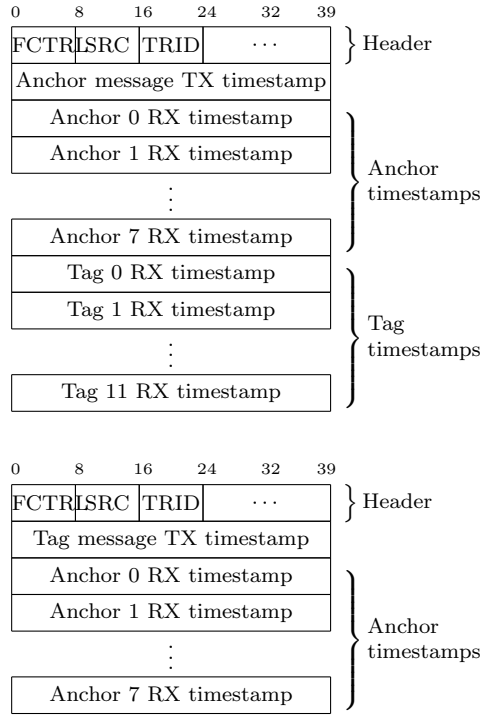


Figure 3.2: The message format of the anchor and tag messages. (top: anchor message, bottom: tag message). FCTRL: Frame control, LSRC: Sender ID, TRID: Message transaction ID.

of the actual message and also a couple of signal receive timestamps. Anchors transmit the received timestamps of the last messages from anchors and tags, while tags transmit only the timestamps of the last received anchor messages.

An example of transmitted messages among four anchors and two tags is plotted in Figure 3.3. In the system, transmission is called a TX event (plotted by black dots), and reception is called an RX event (shown at the end of the arrows). An effective ranging happens between superframes plotted with solid and dotted gray backgrounds since all the timestamps are readily available at the end of each superframe. It can be seen in the figure that every device transmits exactly one message in a superframe.

Comparing the proposed messaging framework against the TWR with grouped messages over the same architecture (see Figure 3.4) shows that localization of two tags requires 12 message slots, while the proposed solution (see Figure 3.3) achieves the same result with 6 slots. Although the lengths of messages are considerably bigger in the framework, the necessary number of message slots is proportional to the number of anchors and tags, while in classical TWR systems, the required number of slots is proportional to the (much higher) number of ranging relations.

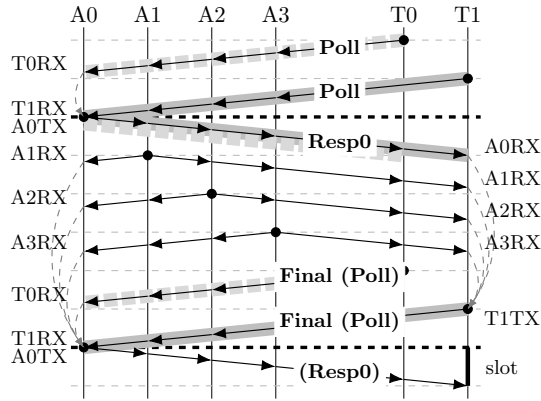


Figure 3.3: TWR over messaging framework. Example of messages among four anchors and two tags. Each black dot represents a message transmission, while the end of the arrows shows the message reception. One TWR ranging transactions of Tag 1 and 0 are plotted with solid and dotted gray backgrounds. Dark dashed lines show the bounds of superframes.

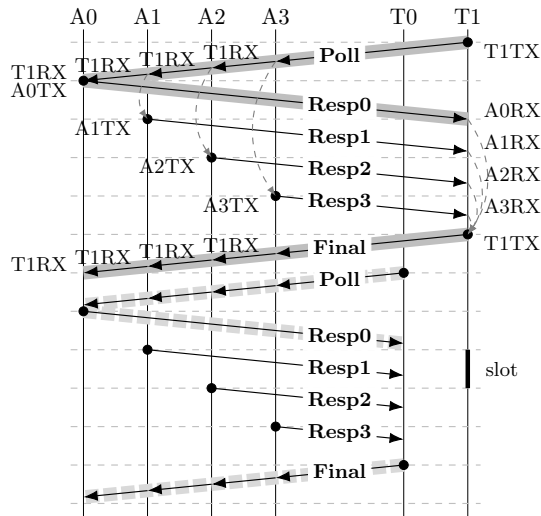


Figure 3.4: TWR with grouped Poll and Final messages. Localization of two tags over the four-anchor architecture requires twelve messages, which number significantly increases with the size of the localization architecture.

3.1.2 Analysis of ranging and localization rate

The main advantage of the proposed solution is that the proposed system provides increased capacity for ranging and localization over the same radio resources. Two measures are computed for both the classical and the novel methods: the *ranging rate* shows the maximum number of ranges (i.e. distance between two devices) that can be measured by the system in a time unit (typically one second), while *localization rate* stands for the same in terms of the maximum number of locations.

Define $\tau(l)$ as the air time of an l bytes length **UWB** packet (see (3.1)). The **TWR** slot consists of a group poll (13 bytes), four responses (15 bytes each), and a final message (65 bytes), and one slot makes four ranging (length are estimations and are typical values). The β ranging rate (for four tags, hence the 4 in the numerator) is

$$\beta_{\text{classic}}^{\text{ranging}} = \frac{4}{\tau(13) + 4 \cdot \tau(15) + \tau(65)} \quad (3.2)$$

The novel method is based on a superframe, which consists of the tag part and the anchor part. The message format is defined in Figure 3.2. So, the superframe rate in one second is

$$\beta_{\text{superframe}} = \frac{1}{c_a \tau(10 + 5c_a + 5c_t) + c_t \tau(10 + 5c_a)} \quad (3.3)$$

where c_t and c_a are the counts of tags and anchors, respectively. From this, the ranging count with anchors and without anchors are:

$$\beta_{\text{novel}}^{\text{ranging,anchors}} = \left[c_a c_t + \binom{c_a}{2} \right] \beta_{\text{superframe}} \quad (3.4a)$$

$$\beta_{\text{novel}}^{\text{ranging}} = c_a c_t \beta_{\text{superframe}} \quad (3.4b)$$

From this, the localization speed can be calculated. For the classic method, supposing four ranging for one location:

$$\beta_{\text{classic}}^{\text{loc}} = \frac{1}{\tau(13) + 4 \cdot \tau(15) + \tau(65)} \quad (3.5)$$

The novel method locates all the tags in one superframe, potentially using all the ranging measurements per anchor and tag pairs. The result is

$$\beta_{\text{novel}}^{\text{loc}} = c_t \beta_{\text{superframe}} \quad (3.6)$$

To present the results, the calculations were done for 850 kbps data rate, 1024 symbols long preamble, 64 MHz **PRF** frequency, and – in the novel method – for eight anchors. This setting is robust enough for a wide variety of use cases. Also, no guard times and no auxiliary frames are involved. For the classic method, both the ranging rate and the localization rate are constant since there are a fixed number of slots in a time unit. Here, while calculating the localization rate, ranging to exactly four anchors were involved. The results for the novel method are slightly more complicated. In one

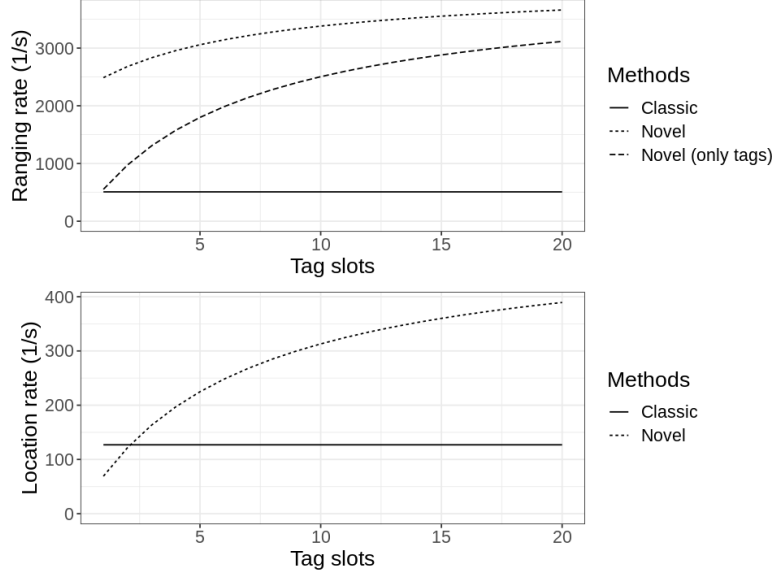


Figure 3.5: Theoretical ranging and localization system capacity against the count of tag slots using 8 installed anchors (higher is better). For details, see text.

superframe, theoretically, all the ranging among the anchors and all the ranging between the anchors and the tags can be performed. Figure 3.5 shows the cases for all the ranges and only the tag ranges. It is important to emphasize that, in the case of localization rate, all the ranges between the anchors and the tags were accounted for, providing more robust localization.

With the same assumption, the – theoretic – limit can be calculated based on (3.5), (3.6) and (3.1) as

$$\lim_{c_t \rightarrow \infty} \frac{\beta_{\text{novel}}^{\text{loc}}}{\beta_{\text{classic}}^{\text{loc}}} = \frac{5961.54n_{\text{pr}} + 1768405.272}{993.59n_{\text{pr}} + 924439.876} \Big|_{n_{\text{pr}}=1024} = 4.054 \quad (3.7)$$

At the limit, using 1024 long preamble, more than 300% increase can be reached in localization rate using the novel method. However, using $c_t = 20$ tags, the increment is reduced to 206%, which is also an impressive result.

The results show a big improvement in localization capacity regarding the novel method. While the ranging rate and localization rate are constant in the case of the classical method, the novel method provides increasing rates (with decreasing increments) while including more tag slots. In almost all situations, the novel method surpasses the ranging and localization rates of the classical method, in some cases providing three times faster localization capacity than the classic solution.

3.1.3 Analysis of ranging error

The novel method shows a big improvement in the speed of localization. However, the method introduces new kinds of errors. Along with the well-known errors presented in TWR (timestamping error, nominal frequency error), two other sources of error arise that require some analysis. One reason behind the classical quick-ranging message exchange is that this way, a roughly static state of the world can be captured, i.e., there is little movement between epochs of message transmissions, and the observed frequency change is negligible. However, in the proposed solution, there can be tens of milliseconds between transmissions. The two most important sources of errors are analyzed in this section: the error resulting from the frequency drift of the oscillators over the superframe duration and the result of movement between transmissions.

Frequency drift error

First, we consider the error caused by the frequency drift between the oscillators of the anchor and the tag. Since the frequency change is relative, we model the situation with two devices, the first with an SKM clock model (see 2.2.2), and the second one with a fixed frequency oscillator running on its nominal frequency. Suppose an $f(t) = f_1 + \Delta \cdot f_1 \cdot t$ linear frequency function of the oscillator of the first device, where f_1 is the nominal frequency of the first device, and Δ is the relative frequency drift to the nominal frequency. The frequency drift during the time-of-flight is ignored, and the transmission of the response message is supposed to be $t = 0$. In this case, the $C_R^{(1)}$ round-trip and $C_D^{(1)}$ delay clock counter values are

$$C_R^{(1)} = \int_{-T_R^{(1)}}^0 f(t) dt = f_1 T_R^{(1)} - f_1 \Delta \frac{(T_R^{(1)})^2}{2} \quad (3.8a)$$

$$C_D^{(1)} = \int_0^{T_D^{(1)}} f(t) dt = f_1 T_D^{(1)} + f_1 \Delta \frac{(T_D^{(1)})^2}{2} \quad (3.8b)$$

Also, suppose some fix f_2 frequency at the second device, the $C_R^{(2)}$ round-trip and $C_D^{(2)}$ delay counter values are

$$C_R^{(2)} = f_2(2T + T_D^{(1)}) \quad C_D^{(2)} = f_2 T_D^{(2)} \quad (3.9)$$

The estimate of time-of-flight (assuming f nominal frequency) using the asymmetric two-way ranging formula (see (2.28)):

$$\hat{T} = \frac{1}{f} \frac{C_R^{(1)} C_R^{(2)} - C_D^{(1)} C_D^{(2)}}{2(C_D^{(2)} + C_R^{(2)})} = \frac{1}{f} (f_1 T - f_1 \epsilon_{TOF}^{drift}) \quad (3.10)$$

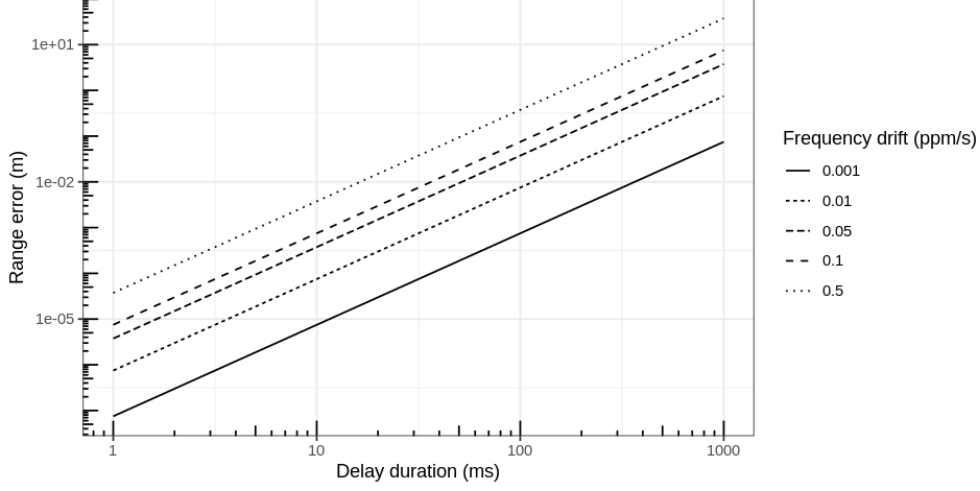


Figure 3.6: The ranging error versus the delay duration at different frequency drift values as a log-log plot. The error is quadratic in the time delay. However, it is in the order of couples of millimeters if the delay is smaller than 100 ms.

where

$$C_R^{(1)}C_R^{(2)} = f_1f_2(4T^2 + 2TT_D^{(2)} + 2TT_D^{(1)} + T_D^{(2)}T_D^{(1)}) - f_1f_2\Delta(2T + T_D^{(1)})\frac{(2T + T_D^{(2)})^2}{2} \quad (3.11a)$$

$$C_D^{(1)}C_D^{(2)} = f_1f_2T_D^{(1)}T_D^{(2)} + f_1f_2\Delta\frac{(T_D^{(1)})^2T_D^{(2)}}{2} \quad (3.11b)$$

$$C_R^{(2)} + C_D^{(2)} = f_2(2T + T_D^{(1)} + T_D^{(2)}) \quad (3.11c)$$

which yields

$$\hat{T} = \frac{1}{f}(f_1T - f_1\epsilon_{TOF}^{\text{drift}}) \quad (3.12)$$

with

$$\epsilon_{TOF}^{\text{drift}} = \Delta\frac{2T(T_D^{(2)})^2 + 4TT_D^{(1)}T_D^{(2)} + T_D^{(1)}(T_D^{(2)})^2 + T_D^{(2)}(T_D^{(1)})^2}{4(2T + T_D^{(1)} + T_D^{(2)})} \quad (3.13)$$

where $T_D^{(1)}$ and $T_D^{(2)}$ are the delay durations at the anchor and the tag, T is the nominal time-of-flight duration, and Δ is the frequency drift. Figure 3.6. shows the error in ranging at different drift values in function of a common delay duration (i.e. $T_D^{(1)} = T_D^{(2)} = T_D$). While the error is quadratic in the time delay however, it is in the order of a couple of millimeters if the delay is smaller than 100 ms, like in our system.

Movement error

Consider the error caused by physical movement between message transmissions. Suppose some linear movement $d(t) = d_0 + v \cdot t$, where d_0 is the distance at the response message (and also the moment of $t = 0$) and v is the speed of the movement. Then the time-of-flight values are:

$$T_p = T - \frac{vT_D^{(2)}}{c} \quad T_f = T + \frac{vT_R^{(2)}}{c} \quad (3.14)$$

where T_p , T and T_f are the time-of-flight of the poll, response and the final message, respectively. Assuming f_1 and f_2 frequencies at the devices, the clock counter values are

$$C_R^{(1)} = f_1(T_p + T_D^{(2)} + T) \quad (3.15a)$$

$$C_R^{(2)} = f_2(T + T_D^{(1)} + T_f) \quad (3.15b)$$

Similarly

$$C_D^{(1)} = f_1T_D^{(1)} \quad C_D^{(2)} = f_2T_D^{(2)} \quad (3.16)$$

This yields

$$\hat{T} = \frac{1}{f} \frac{C_R^{(1)}C_R^{(2)} - C_D^{(1)}C_D^{(2)}}{2(C_D^{(1)} + C_R^{(1)})} = \frac{1}{f}(f_2T + f_2\epsilon_{TOF}^{move}) \quad (3.17)$$

with

$$\epsilon_{TOF}^{move} = \frac{2T(\frac{v}{c})T_D^{(1)} - \frac{v^2}{c^2}T_D^{(1)}T_D^{(2)}}{2(2T + (1 - \frac{v}{c})T_D^{(2)} + T_D^{(1)})} \quad (3.18)$$

where $T_D^{(1)}$ and $T_D^{(2)}$ are the delay durations at the anchor and the tag, T is the nominal time-of-flight duration and c is the speed of the electromagnetic wave. It is also worth noticing that the error is only dependent on the distance at the moment of transmission, not the actual $d(t)$ function. Figure 3.7. shows the error at different delay durations in the function of the speed of the movement.

The error seems unintuitively small. However, one can observe that the TWR consists of two one-way ranging parts, so the result tends to the time-of-flight of the middle message. To avoid a symmetric situation (i.e., the average of the time-of-flight of the poll and final message is exactly the response time-of-flight), the $T_D^{(1)}$ and $T_D^{(2)}$ values are not always equal in Figure 3.7. Note that movement error also arises in classical ranging methods among the ranging measurements to different anchors.

3.2 Evaluation

The evaluation of the proposed solution is based on the well-known Decawave DWM1001 module [83], installed in a room of an office building. The distances of the anchors were measured by a laser distance measurement tool, which can be seen in Table 3.2. The

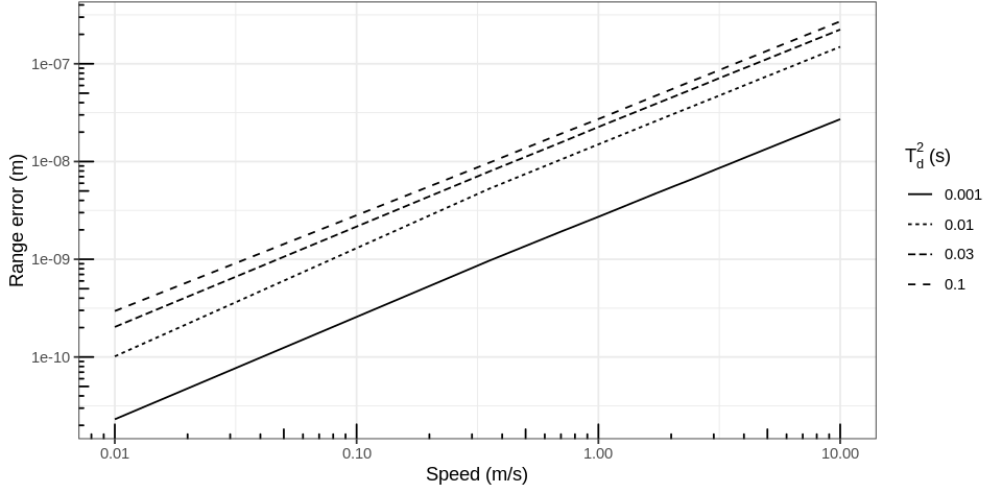


Figure 3.7: The ranging error versus the movement speed at different $T_D^{(2)}$ delay duration as a log-log plot. The delay of $T_D^{(1)}$ is fixed at 10 ms. The error is close to linear in terms of the speed and is negligible.

heights of the anchors and the tag are 1.5 meters. The coordinates of the devices are restored by multidimensional scaling, which results in the arrangement seen in Figure 3.8. The reconstruction error proved to be as low as 5.4 mm. The 6 different anchors provide 6 ranging constellations.

Table 3.2: The distances of anchors in the arrangement used for evaluation (in meters)

	A0	A1	A2	A3	A4	A5
A1	2.50					
A2	4.29	1.79				
A3	7.01	5.80	5.52			
A4	5.82	5.58	6.07	2.50		
A5	5.53	6.07	7.01	4.31	1.82	
T	4.55	4.72	5.58	3.52	1.35	1.38

Both the classical method and the novel method were implemented. The classical method uses 4 anchors in one ranging group, while the novel method uses the superframe shown above with 8 anchor slots and 12 tag slots resulting in a superframe length of 100 ms. For ranging calculations, the asymmetric **TWR** formula was applied.

Previously, it was shown that the main additional source of error is the frequency drift of the crystal oscillators, so it is essential to prove that there are drifts present while evaluating the proposed solution. While this is hard to measure, one approximation is to infer the relative frequency values from the measured timestamps using the clock of a tag as a reference. This way, the frequencies of the anchors' clocks can be calculated

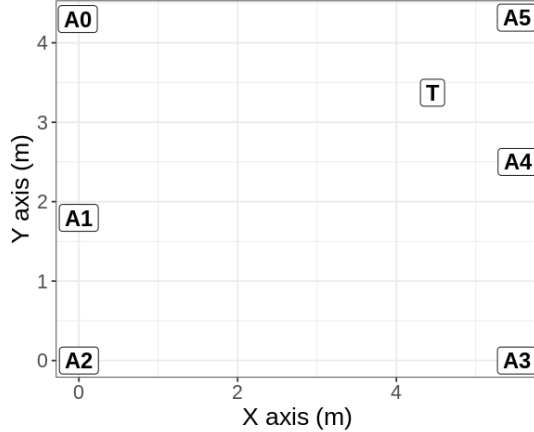


Figure 3.8: The arrangement of the devices during evaluation. There are 6 anchors installed and 1 tag placed around the room. The coordinates are in meters.

relative to the frequency of the tag’s clock (see Figure 3.9). Fortunately, as we saw, the error depends only on the *relative* frequency drift (ignoring the error between the nominal and actual frequency).

The ranging results can be seen in Figure 3.10, where the distributions of the ranging values are plotted as violin plots of classic and novel methods pairwise. Table 3.3. shows the summary statistics of the distributions, the median and the standard deviation. Median was selected as it is less sensitive to outliers compared to the mean. Both the figure and the table show that the distributions are very similar, the median error is in the order of centimeters. However, between different measurements, it is hard to provide the exact same environment, so changes in channel propagation might slightly affect the result. Also, the standard deviations from the ground truth values are systematic errors arising from the lack of precise ground truth values.

In conclusion, both the classic and novel methods offer nearly identical localization capabilities, but the novel method is significantly faster, especially when there are many tags. Additionally, the novel method provides more measurements, potentially leading to more accurate localization. However, there are a few drawbacks to the method. Its scalability is limited by the maximum length of UWB messages, although the standard includes extensions to increase packet length, which is sufficient for typical use cases. Additionally, using longer packets increases power consumption, but this can be mitigated by the TDMA scheme inherent in the novel method, which allows receivers to be disabled during irrelevant time slots.

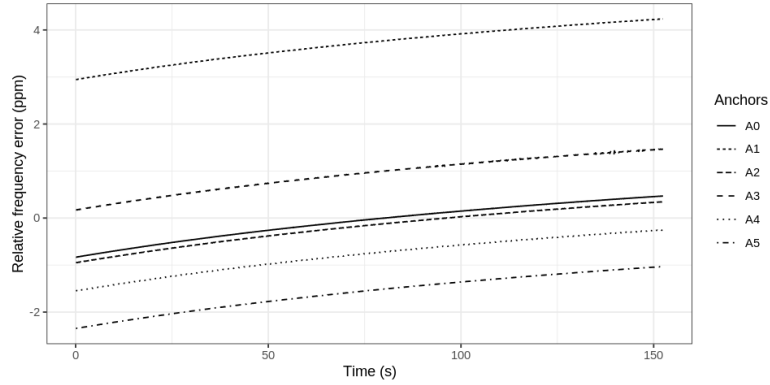


Figure 3.9: The frequency error in the novel measurement relative to the tag’s clock. It is easy to see that the relative frequencies of the anchors are different, and there is a (modest) drift in each relative frequency. While these are moving together, it might be caused by the frequency drift of the tag itself.

Table 3.3: Results of ranging evaluation (in meters)

	Classic		Novel		Abs. Error
	Median	Std. Dev.	Median	Std. Dev.	
A0	-0.0088	0.0348	-0.0157	0.0248	0.0069
A1	0.0308	0.0115	0.0661	0.0169	0.0353
A2	0.1165	0.0235	0.1176	0.0202	0.0011
A3	0.0929	0.0314	0.1005	0.0305	0.0076
A4	0.0114	0.0198	0.0356	0.0178	0.0242
A5	0.0240	0.0185	-0.0223	0.0218	0.0464

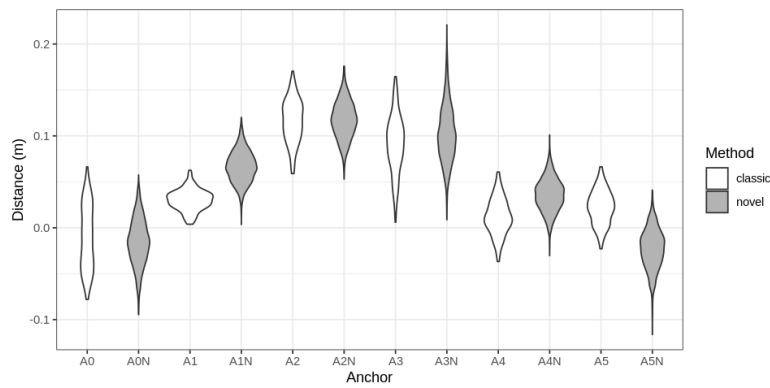


Figure 3.10: Results of ranging evaluation. The distribution of the ranging results is plotted for each anchor, where the N postfix indicates the novel method (and also filled with gray). The ground truth distances are calculated from the arrangement and are subtracted.

3.3 Thesis summary

Thesis Group 1 – I have shown that increasing the effective data rate in Ultra Wide Band systems by using a special messaging scheme with larger messages results in a higher ranging rate and more frequently received RX timestamps. I have designed a novel, scalable time-division messaging scheme that provides higher synchronization, ranging and localization rate in ultra wide band systems. I have shown, that the increment in the ranging rate is accompanied by a tiny increment in the ranging error. [C1]

Thesis 1.1 – I have shown that the increment in effective data rate can be exploited to raise the localization rate (i.e. computed locations per second) using a specific framing structure. The localization rate can increase up to 3 times compared to the classical ranging methods. [C1]

Thesis 1.2 – I have shown that the ranging error in a system presented above slightly increases the ranging error through the frequency drift between the transceivers and the movement of the transceivers. I have derived analytical approximations for the ranging error given a simple skew clock model and linear movements. I have shown that the ranging error can be neglected in typical applications. [C1]

4

Synchronizing PTP clocks using UWB

The IEEE 1588-2019 standard [11] addresses the topic of sub-microsecond accurate clock synchronization over a communication network; however, in most cases, the accurate clock synchronization solutions are implemented on a wired network with PTP. There are many wireless implementations based on [Global Positioning System \(GPS\)](#) synchronization, which are also capable of sub-microsecond clock synchronization, but they have significant limitations for indoor applications. Therefore, wired PTP solutions are favored for indoor clock synchronization applications. Still, there are advantages of certain wireless clock synchronization solutions in an indoor environment for validation and wireless master clock purposes.

This chapter proposes a method to synchronize local PTP clocks by means of wireless [UWB](#) communication. While synchronizing UWB clocks is a widely studied topic, using UWB communication for synchronization of PTP-enabled clocks of Ethernet devices is a rarely researched area, without few to no actual results published to the authors's best knowledge. The main advantage of the presented method is that the PTP clocks (as defined in IEEE 1588 standards) can then readily be used as master clocks in a PTP network.

Even though wired solutions are the main direction of PTP implementations, there are existing solutions and research in the wireless PTP domain. Óscar et al. [1] proposes a timestamping method for time synchronization over [Wireless Local Area Network \(WLAN\)](#) standard conditions. Their paper presents several simulations using the IEEE 802.11n physical layer and four wireless time-dispersive and time-variant channel models. They have also performed validation experiments using an 802.11g modem implemented over a high-performance Software Defined Radio hardware platform. Inaki et al. [84] present an IEEE 802.1AS clock synchronization performance evaluation of an integrated wired-wireless TSN architecture. Wireless TSN is expected to be integrated with Eth-

ernet TSN to create large-scale wired–wireless [TSN](#) networks. Their paper presents two hardware architectures to enable clock synchronization distribution among the network domains. Another Wireless PTP implementation is presented in [85]. This paper evaluates the performance of the over-the-air time synchronization mechanism, which has been proposed in 3GPP Release 16 [86]. The paper analyses the accuracy of time synchronization through the boundary clock approach in the presence of clock drift and different air-interface timing errors related to reference time indication. The paper also investigates the frequency and scalability aspects of over-the-air time synchronization. The performance evaluation reveals the conditions required for accuracy of $1\mu\text{s}$ or below in TSN time synchronization.

In the case of IEEE 802.11 (commonly known as [WiFi](#)), the software-based PTP solutions are capable of reaching time synchronization error in the order of 10 microseconds using wireless links [87,88]. However, more precise synchronization can be achieved using the WiFi receiver information. Since the IEEE 802.11-2006 standard, the protocol supports the [fine timing measurement \(FTM\)](#) for ranging purposes, which can be used for precise time-of-flight measurement [89]. The IEEE 802.11AS standard [27] also mentions the WiFi as the possibly media-dependent layer; however, available implementations are lagging behind.

There are a couple of explicit hardware solutions or circuits to help the implementation of the PTP protocol for Ethernet-based (IEEE 802.3) networks. However, wireless products do not provide timestamping features or hide the information that is needed to calculate accurate RX and TX timestamps. Furthermore, the propagation characteristics of the radio channel – especially in non-line-of-sight situations – generate jitter in receive timestamps. Fortunately, UWB is perfectly suitable for timestamp reception and transmission of packets since indoor localization requires sub-nanosecond accuracy to provide location in centimeter precision. UWB is applied in various use cases [90–92], but its main application area is indoor, centimeter-based localization using ranging techniques. UWB technology is also applied in time synchronization and synthonization use-cases [93–95], mainly for localization purposes (e.g. in time-difference-of-arrival systems). Marcelo et al. [96] achieves 5-ns RMS results in a UWB network, while the Wicsync solution reports errors below 3 nanoseconds [97]. However, all the solutions synchronize the clocks in the UWB clock domain, which avoids the need for the clock domain change. UWB applies very short pulses, which help to estimate the [channel impulse response \(CIR\)](#) to extract the first component arrived, hence providing precise timestamps for the first (and hopefully the line-of-sight) component. The IEEE 802.14.5 standard [14] specifies the ranging counter as the clock for timestamping events, which has a resolution of around 15 picoseconds; however, the timestamping accuracy is in the order of nanoseconds. Naturally, the proposed solution uses the high-precision timestamping feature of the UWB device to accurately timestamp the PTP messages on the UWB radio link.

4.1 Use cases

There are various use cases for wireless PTP synchronization. However, there are two target use-cases presented in this subsection, as shown in Figure 4.1:

- wireless synchronization between master and slave PTP devices and
- wireless PPS signal transmitter.

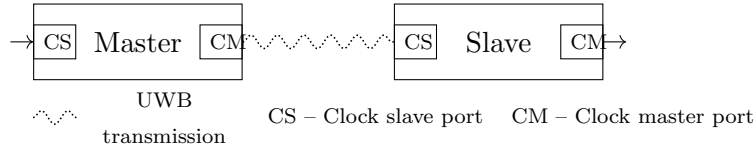
The first one aims to provide means to synchronize a local PTP master clock to a remote grand master clock using UWB communication, which can be applied in situations, e.g., where wiring would be hard and expensive between two subnetworks or there are no transparent clocks between the subnetworks.

The other use case is related to the industrial standard to validate clock synchronization accuracy in PTP networks, which is based on pulse per second (PPS) signals. During the validation, the PPS signal of the clock of the synchronized device can be compared to the PPS signal of a reference clock, from which the synchronization accuracy can be measured. For a grand master reference, GPS is widely applied; however, it is prevalent that GPS cannot be an option in an indoor environment, while the wired network's PPS signal transmission requires additional cabling and network configuration. Such challenges open the path for wireless PTP validation where there is no need for additional cabling and offer a flexible solution. Besides validation, wireless PTP can be utilized as a master clock to eliminate non-transparent PTP devices from the PTP network. Also, there is no need for cabling; existing network devices do not have to be replaced, and it can be significantly cheaper than wired PTP system implementations.

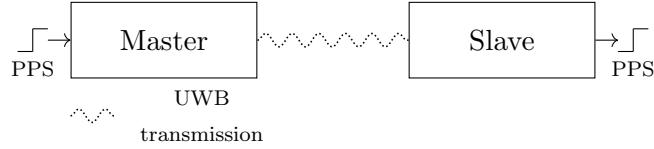
In the proposed solution, the focus is on the second use-case where the aim is to transmit a PPS signal from the master to the slave device, e.g., for diagnostic purposes to check the synchronization between remote devices without transmitting the PPS signal through the wire.

4.2 Synchronizing local clock domains

The main issue regarding the synchronization of PTP clocks using UWB communication is the different clock domains of the UWB transmitter and the PTP clock. While this could be overcome in actual hardware, there are two reasons why this cannot be done. On the one hand, it is really hard to fit the clock frequency of the Ethernet-based PTP standard (IEEE 1588) and the clock frequency of the UWB physical layer (IEEE 802.15.4) using different frequencies without common divisors. On the other hand, actual implementations are separate PTP devices (mainly system on a chip (SoC) solutions) and UWB transceivers. Note that the clock domain in this section means the different local clock speeds of the SoC and the UWB transmitter and not the clock domain concept of PTP! Figure 4.2 shows a typical UWB transmitter solution with its clock domains, where the UWB transmitter is controlled by a general SoC. While UWB communication typically requires a quality clock signal (i.e. UWB is very sensitive to the phase noise),



(a) Use-case for a wireless boundary clock, where the Slave device synchronizes to a Master device through UWB, and acts as a master device to other slaves.



(b) Wireless PPS signal transmitter.

Figure 4.1: The target use-cases for the UWB-based PTP synchronization.

SoC microcontrollers are satisfied with cheap oscillators. To synchronize the PTP clock, the timestamp of RX and TX events needs to be timestamped in the PTP clock domain. The question is, if the exact timestamp of some event is known in the clock domain of the UWB transmitter, how can the timestamp of the same event be calculated in the clock domain of the PTP clock?

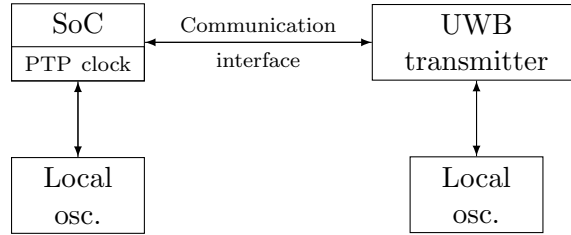


Figure 4.2: Typical ultra wide-band hardware solution using different clock domains on the UWB transmitter and the SoC

Having two ideal clocks in different clock domains, the general clock model can be written for both clocks:

$$C^{\text{SoC}}(t) = \int_0^t f^{\text{SoC}}(\tau) d\tau + o^{\text{SoC}} \quad (4.1a)$$

$$C^{\text{UWB}}(t) = \int_0^t f^{\text{UWB}}(\tau) d\tau + o^{\text{UWB}} \quad (4.1b)$$

Here, C is the actual clock counter value, $f(t)$ is the actual frequency, o is the offset of the clock, and t is the wall-clock time. Using crystal oscillators or even **TCXOs**, it is straightforward that the frequency drift between f^{SoC} and f^{UWB} accumulates very fast. One solution is to use expensive **OCXO** oscillators with low-frequency error (i.e., 1 ppb).

However, it is easier to use the same clock for the SoC and the UWB transmitter, i.e., $f^{\text{SoC}}(t) = f^{\text{UWB}}(t)$. In this case, subtracting (4.1b) from (4.1a):

$$C^{\text{SoC}}(t) = C^{\text{UWB}}(t) + (o^{\text{SoC}} - o^{\text{UWB}}) = C^{\text{UWB}}(t) + \Delta o \quad (4.2)$$

, where Δo is the offset between the SoC and the UWB chip clocks.

However, even with a common clock source, the offsets in each clock domain cannot be controlled since the PLLs and other circuits make the clock initialization somewhat stochastic, so Δo can change at each reset. Consider that we would like to synchronize the clock through UWB communication using the classic PTP message exchange (see Figure 2.6). Denote Δo_m the offset at the master and Δo_s the offset at the slave. Define

$$\begin{aligned} t_1 &= C_1^{\text{UWB}}(t) + \Delta o_m \\ t_2 &= C_2^{\text{UWB}}(t) + \Delta o_s \\ t_3 &= C_3^{\text{UWB}}(t) + \Delta o_s \\ t_4 &= C_4^{\text{UWB}}(t) + \Delta o_m \end{aligned} \quad (4.3)$$

Combine (4.3) and (2.20), which yields

$$\Delta t = C_2^{\text{UWB}}(t) - C_1^{\text{UWB}}(t) + \Delta o_s - \Delta o_m - \frac{C_2^{\text{UWB}}(t) - C_1^{\text{UWB}}(t) + C_4^{\text{UWB}}(t) - C_3^{\text{UWB}}(t)}{2}$$

Unfortunately, the difference $\Delta o_s - \Delta o_m$ is unknown and dependent on the initial offset of the clock, which is mainly random. Ideally, Δo_m should equal Δo_s , and there shall be some process providing constant Δo offsets. To achieve this, some kind of synchronization facility needs to be provided by the UWB transmitter, e.g., the DW1000 chip is able to reset the internal timebase triggered by an external pulse. A concrete example is presented in the later sections; however, the exact method of synchronization depends on the actual hardware capabilities used in the implementation.

4.3 Synchronizing local PTP clock using ultra wide band radio

The implementation is based on a STM32F407 SoC, which is a powerful, widely used ARM Cortex M4-based device used in embedded systems [98]. The chip has an Ethernet MAC with a built-in frequency tunable PTP clock, which is able to timestamp RX and TX events on the MAC layer. The UWB communication is provided by the widely used Qorvo DW1000 chip, which has advanced timestamping and external synchronization capabilities. The evaluation devices are designed by the author and can be seen in Figure 4.3.

Figure 4.4 shows the connections between the components. The clock is generated by a 38.4 MHz temperature-controlled oscillator (TCXO) with a very low phase noise (-132 dBc/Hz at 1 kHz). The clock is distributed between the components by a PL133

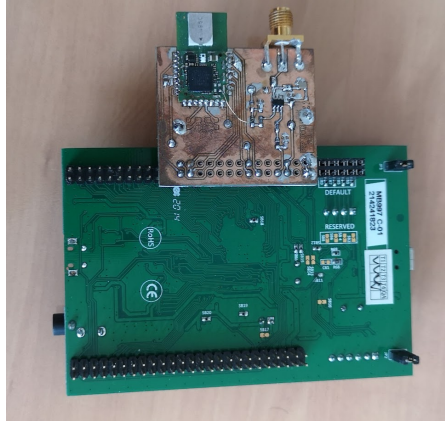


Figure 4.3: The evaluation device is based on the STM32F407 evaluation board and extended by a user-made board containing the DWM1000 module and the clock circuits with an optional SMA connector for the external clock signal.

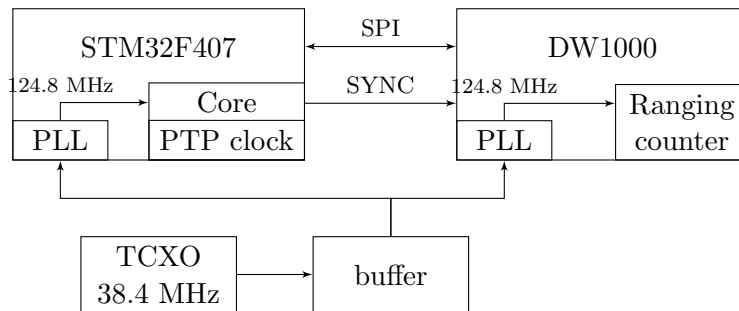


Figure 4.4: The synchronization circuit for evaluating the clock synchronization algorithm. The SoC (STM32F407) and DW1000 UWB transmitter communicate through an SPI interface, and clock synchronization is performed by the Sync signal. The ranging counter and the SoC system clock (and also the PTP clock) are running at 124.8 MHz.

clock distributor IC, which also provides low additive phase noise (in the order of femtoseconds). The DW1000 chip uses a PLL to generate the 124.8 MHz signal to drive the ranging counter. The ranging counter is 40 bits wide, providing a resolution of 15 picoseconds. However, the actual timestamping happens on the 124.8 MHz system clock (i.e. the last 9 bits are zeros), and the chip uses a special algorithm based on the channel impulse response to fine-tune the timestamp [81]. While the STM32F407 SoC can operate up to 168 MHz, to simplify implementation, the STM32F407 SoC is also programmed to generate a $f_{\text{SYS}} = 124.8$ MHz signal, and the PTP clock and also the Cortex core are driven by this clock.

4.3.1 The hardware PTP clock

The hardware PTP clock ([PTP hardware clock \(PHC\)](#)) consists of a 32-bit wide second register and a 31-bit wide subsecond register, allowing a resolution of 0.46 nanoseconds [99]. While the PTP clock is driven by the 124.8 MHz clock, the counter is virtually running somewhat slower. The frequency of the counter can be tuned by a so-called addend register (C_{ADDEND}) shown in Figure 4.5, which is added to the accumulator register at each clock cycle, effectively changing the clock frequency as

$$f_{\text{PTP}} = \xi \cdot f_{\text{SYS}} \cdot \tau \quad \xi = \frac{C_{\text{ADDEND}}}{2^{32}} \quad \tau = \frac{C_{\text{SS}}}{2^{31}} \quad (4.4)$$

, where C_{SS} is the value of the 31-bits subsecond register, τ is the resolution of the PTP clock, f_{SYS} is the SoC frequency and ξ is the frequency tune coefficient.

Changing the addend register, the clock frequency changes. To select the correct values for the addend and the subsecond register, one should compromise the resolution of the PTP clock and the limits of the clock frequency tuning. While the former provides better accuracy, the latter limits the control signal from the clock controller (e.g. PI controller).

4.3.2 Synchronization with the DW1000 chip

The main issue in synchronization is that the packet receive and transmit time instants shall be known by the PTP clock. Typically, the PTP clock is integrated into the hardware that receives and transmits the packets. Also, the DW1000 UWB chip provides timestamps using its own counter (with its own offset Δo). However, the clock of the DW1000 chip cannot be tuned, so it is not suitable for use as a PTP clock. External synchronization facilities shall be used to synchronize a PTP clock outside of the DW1000 clock domain. External synchronization is implemented by a special SYNC signal in the DW1000 chip. There are two modes used: [one-shot transmit synchronization mode \(OSTS\)](#) and the [one-shot timebase reset \(OSTR\)](#).

One-shot timebase reset mode allows a reset to be applied to the timebase counter used for timestamping in DW1000 at a deterministic and predictable time relative to a synchronization event (i.e., the SYNC pin). The DW1000 chip will reset the counter at a repeatable time to typically less than 100ps variation. Besides OSTR mode, one-shot

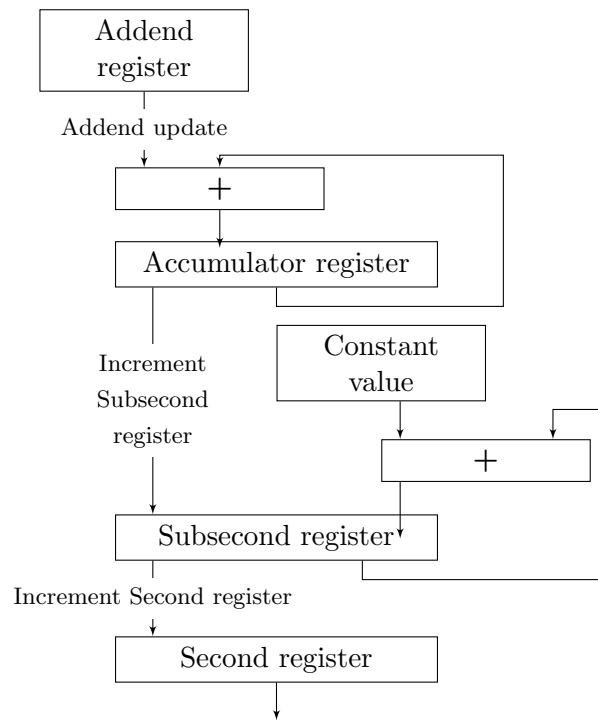


Figure 4.5: The operation of the PTP clock. The PTP clock consists of a second and a subsecond register. The subsecond is incremented by a constant when the accumulator register is overflowed. [99]

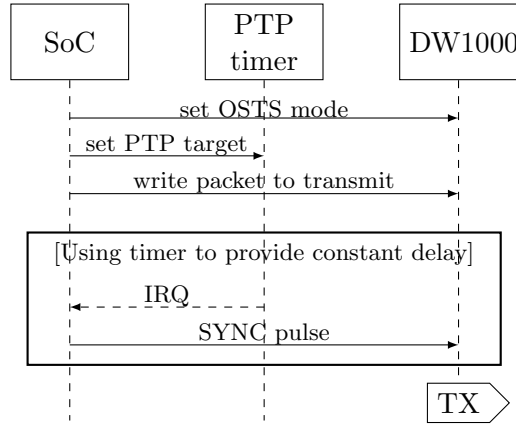


Figure 4.6: The method of TX synchronization. The transmission starts after the SYNC pulse with a constant delay.

transmit synchronization mode provides for the transmission of a frame at a well-defined time (typically 12 ps) relative to the assertion of the SYNC pin of the DW1000 chip. To learn more of the mechanism of OSTS and OSTR mode, the DW1000 user manual has a detailed description [81].

Transmission synchronization

For scheduled transmission, *OSTS* mode is used. For transmitting a packet at a specified moment in the future, the PTP clock target is set to give an interrupt (see Figure 4.6). However, interrupts on Cortex M4 are not ensured to be real-time, so it introduces a jitter of a couple of clock cycles. To avoid the jitter, the PTP interrupt triggers a timer, which constructs a fixed-sized SYNC pulse in constant time. At the rising edge of the SYNC pulse, the DW1000 is guaranteed to start transmission in a fixed time delay. Notice that the transmission timestamp can be included in the message, so a one-step method can be used, which does not need to send a *Follow-Up* message.

Receive synchronization

Receive synchronization has to provide a constant Δo offset. The DW1000 chip is able to reset its timebase against a rising edge of the SYNC signal using the external synchronization mode OSTR. When setting the SYNC signal, the PTP clock shall be read and stored as t_{SYNC} (see Figure 4.7). In a Cortex M4 processor, the two 32-bit registers of the PTP clock and the GPIO change cannot be performed simultaneously; however, it can be done with a constant delay with disabled interrupts.

Later on, at reception, the DW1000 timestamp can be read out, and the PTP timestamp for the received event can be calculated as

$$t_{\text{PTP}} = t_{\text{SYNC}} + \frac{C_{\text{DW}}}{29} \cdot \xi \cdot f_{\text{SYS}} \cdot \tau \quad (4.5)$$

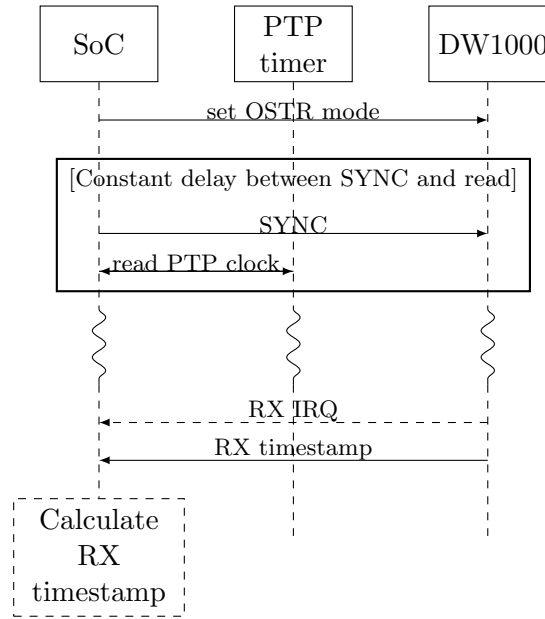


Figure 4.7: The method of RX synchronization. The transmission starts after the SYNC pulse with a constant delay.

, where C_{DW} is the 40 bit DW1000 timestamp. The division with 2^9 is performed to get the timestamp by the 124.8 MHz frequency because the resolution of the DW1000 timestamp is $1/(499.2 * 128 \text{ MHz})$.

4.3.3 Timestamping errors

In the implementation, there are a couple of potential errors or jitter sources.

PTP clock resolution

One of the most basic jitter sources is the resolution and frequency of the PTP clock. The limited resolution (see (4.4)) of the PTP clock results in a classical sampling noise; however, the situation is worse as the frequency tuning by the addend register deviates from the sampling points. Increasing the effective frequency of the PTP clock results in a decrement in the sampling noise.

Wireless propagation

The stochastic behavior of wireless propagation results in timestamping errors, i.e. the first component that arrived is wrongly detected and timestamped. To avoid this issue, UWB devices observe the channel impulse response to refine the timestamp, acquiring subnanosecond accuracy of timestamping. However, pure **NLOS** propagation can add a higher timestamp offset, which is hard to detect.

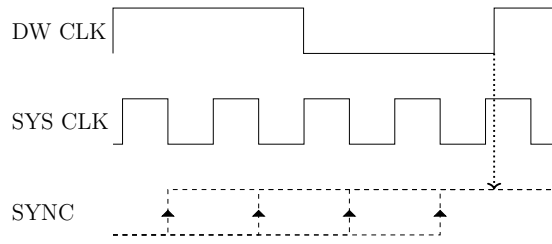


Figure 4.8: Jitter resulting from clock domain change. The SYNC signal is sampled at the rising edge of the DW clock but driven by the SYS clock. (Note: the SYS CLK can be at any phase.)

Clock domain synchronization error

The main source of error is the clock domain switch between the SoC and the UWB transmitter. The SYNC signal of the DW1000 chip is sampled at the rising edge of the 38.4 MHz clock; however, SoC runs on a faster clock, 124.8 MHz. The SoC toggles the SYNC signal on its own clock, which can result in some jitter since SYNC signals at different rising edges of the SoC clock are sampled at the same rising edge of the DW CLK (see Figure 4.8). This error can introduce around 26 nanoseconds of jitter or uncertainty in timestamping due to the 38.4 MHz sampling clock.

4.3.4 Results

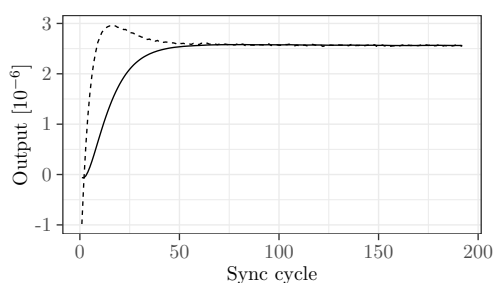
The evaluation is based on the implemented synchronization method presented in Subsection 4.3.2. For evaluation, two devices were used: one is a free-running master device, and the other one is the slave device synchronizing with the master device. The control of the clock is implemented in the same way as the PTPd project, with [Finite Impulse Response \(FIR\)](#) filters on the offset from the master path but without the [Infinite Impulse Response \(IIR\)](#) filter on the time of flight (see [100]). The setting of the arrangement is summarized in Table 4.1. Please note that the parameters of the PI controller (K_i , K_p) are not optimal in any sense; it is selected to be stable and conservative since, for our experiment, the actual PI parameters can be neglected.

After a cold start, the PI controller starts to converge to the synchronous state, as shown in Figure 4.9. The figure shows that the error signal's steady state converges to zero in the order of nanoseconds, and the control signal for the PTP clock converges to a steady state. The control signal has a small overshoot, and the convergence happens in 70-80 sync cycles corresponding to 35-40 seconds. The parameters of the PI controller were selected so that the system would be stable with a high integrator coefficient. The high integrator coefficient helps to reach a stable state relatively fast and removes static error. However, this work does not target the design method of the optimal controller for transient behavior; only concentrates on the steady state.

After reaching the steady state, 1000 measurements were made by a DSO-X 3054A oscilloscope, using the Pulse Delay Measurement function to analyze and record the

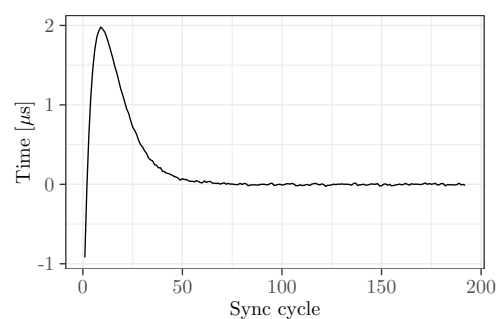
Table 4.1: The settings used for evaluation to reproduce the results. (For reference, see DW1000 manual [81] and STM32F manual [101].)

Subsecond update	0x14
Addend	0xDC41363A
Sync period	500 ms
SYSClk	124.8 MHz
K_i (PI)	15
K_p (PI)	1
FIR	[0.5,0.5]
UWB channel	5
UWB bitrate	850 kbps
UWB PRF	64 MHz
UWB preamble syms	1024



— Integrator signal - Control signal

(a) The PI controller control signal and the output of the integrator



(b) The error signal of the PI controller

Figure 4.9: The error signal and the output of the PI controller of the clock servo. The time axis shows the synchronization cycles, where one cycle is 500 ms. It can be clearly seen that the error signal is in the order of nanoseconds.

delay between the PPS signals of the master and slave PTP clock (see Figure 4.10). The result can be seen in Figure 4.11 as a histogram. It can be noticed that the error between the two PPS signals does not exceed 30 nanoseconds in absolute value; however, the distribution is skewed, and there are fewer negative errors than positive errors. 90 percent of the absolute errors are below 20 nanoseconds, while 70 percent of the absolute errors are below 10 nanoseconds.

The skewness of the distribution assumes that the timestamping error is asymmetric, i.e. it is not exactly the same in the case of master-to-slave and slave-to-master. The errors are likely to originate from a couple of sources presented before:

1. The error resulting from clock domain change.
2. The PTP clock resolution.

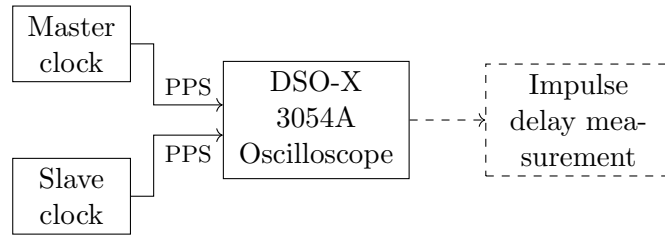
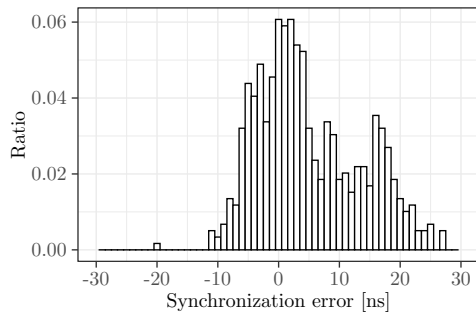
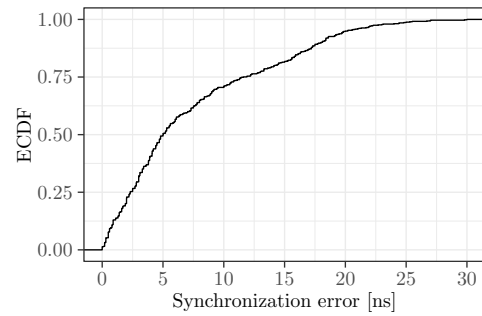


Figure 4.10: The measurement setup of the clock synchronization error between the master and slave devices.



(a) Histogram of the measured errors



(b) The empirical cumulative distribution plot of the absolute error

Figure 4.11: Distribution of the error measured between the two PPS signals. The histogram is based on 1000 sync cycle after the controller has locked.

3. Multipath propagation.

It is worth noticing that the magnitude of the error can be explained by the clock domain change error in itself and is even well below the very strict requirements of 5G System synchronicity budget (i.e. 700 ns) [102].

4.4 Thesis summary

Thesis Group 2 – I proposed, designed, and implemented an ultra wide band-based method to synchronize hardware PTP clocks with high precision. I have shown that the main challenge is the chip-level clock domain change between the PTP clock and the UWB transmitter. To verify the proposed solution for the clock domain change, I designed and implemented a DW1000 chip-based hardware with a PI controller to follow the remote clock. An experiment in an indoor environment shows that the accuracy of the synchronization is in the order of 10 nanoseconds, which is well below the requirements of the strictest synchronicity budget of the 5G system.

Thesis 2.1 – I have shown that the synchronization of an Ethernet PTP clock to a UWB timestamping clock requires local clock offset synchronization in order to end-to-end synchronize PTP master and slave clocks. I proposed a method to implement clock domain change between a DW1000 chip and an STM PTP configurable clock. [J1]

Thesis 2.2 – I evaluated the proposed method of clock domain change between an STM PTP clock and a DW1000 chip using end-to-end PPS clock offset measurement with a PI controller-based clock servo in an indoor environment. I presented that the synchronization accuracy is in the order of 10 nanoseconds. [J1]

5

Measurement distribution adaptive synthonization in UWB communication systems

Section 2.4 has shown that acquired receive timestamp in ultra wide band systems itself is insufficient for localization, e.g. two-way ranging makes use of a couple of transmit and receive timestamps [75]. For estimating position, robust solutions nowadays apply complex probabilistic methods to implement ranging or localization, providing means not only to determine the position of the asset but also to estimate the uncertainty of it [103–107]. Also, probabilistic solutions are applied for estimating anchor clock parameters for [time difference of arrival \(TDoA\)](#) techniques, e.g., using Kalman filters like in [108]. Specifically, a couple of papers aim the problem of estimation of receive timestamps in [NLOS](#) conditions [109, 110]. Note that all presented solutions use some explicit (i.e., Bayesian inference, Kalman filters) or implicit (e.g., averaging over measurements) probabilistic inference for estimating the requested quantity from the receive timestamps.

Interestingly enough, inference is always carried out assuming Gaussian distribution on the receive timestamps, e.g. using Kalman filters or extended Kalman filters to estimate range or clock drift implicitly presume Gaussian distribution on the measured values. In ranging and localization techniques, averaging or regression is also carried out implicitly or explicitly, assuming Gaussian distribution. However, no trace of other claims on the distribution can be found in the literature. For instance, it is straightforward, intuitively, that the receive timestamps have some distribution on positive numbers because – except for some outlier timestamping errors – the multipath components are received *after* the ground-truth timestamp. Since multipath must be accounted for, it is tempting to model the arrival of the components alone – without any assumption on the radio channel model – as a Poisson process, resulting in exponential delays. However, it does not mean directly that the distribution of received timestamps has an exponential

distribution. On the other hand, the timestamping process can be examined – combined with a widely known radio channel model – which provides a clear answer about the distribution shape of the receive timestamps.

Furthermore, the radio channel is generally not time-invariant – the channel impulse response can change rapidly, resulting in rapid variations in the quality of receive time estimation. This implicitly means that the distribution of the received timestamps changes with time, leading to varying variances. While continuously estimating the measurement noise is computationally infeasible, some probabilistic methods, like Kalman filtering, do not require the exact description of the measurement noise. Rather, it is a good compromise to adaptively estimate the variance of the measurement noise to provide optimal clock state estimation.

The first part of the chapter investigates the distribution of the UWB receive timestamps in office indoor environments using simulations of the standard propagation model. First, the propagation model of the IEEE 802.15.4 standard is presented using a typical timestamping procedure. The results show some simulation results among the proposed closed-form distributions for describing the receive timestamp probability density function.

After establishing the measurement noise distribution through simulation, Section 5.2 presents the methods used to measure the real measurement noise using ultra wide band communication in three different scenarios. Based on the various noise profiles, Section 5.3 introduces the probability clock model used in filtering-based estimation and proposes an expectation maximization-based estimation of the measurement noise variance.

5.1 Modeling the distribution of receive timestamps

Theoretically, to model the distribution of the receive timestamps in UWB systems, the channel model of the ultra wide band propagation and the timestamping process shall be combined to derive the exact distribution of the receive timestamps. However, while it is preferred to find a closed-form distribution of the receive timestamps, the channel model is complicated enough to perform simulation instead to get some insight into the distribution. In the simulations, samples need to be drawn from the radio channel impulse response, on which the timestamping procedure can be applied.

5.1.1 Channel model

The standard propagation model used in UWB systems is the model proposed in the final report of the IEEE 802.15.4a channel mode [111]. The model is based on the original Saleh-Valenzuela model (SV) [112], although there are some fundamental modifications.

The SV channel model uses a classic ray-tracing model; however, it assumes that the rays arrive in clusters, so the impulse response is a double sum, i.e.

$$h(t) = \sum_{l=0}^L \sum_{k=0}^{K_l} a_{k,l} e^{j\phi_{k,l}} \delta(t - T_l - \tau_{k,l}) \quad (5.1)$$

where L is the number of clusters, K_l is the number of components in the cluster, which is dependent on the actual cluster. The $a_{k,l}$ and $\phi_{k,l}$ are the attenuation and the phase of a component. The T_l is the delay of the l -th cluster, and $\tau_{k,l}$ is the delay of a component. By definition, $\tau_{0,l} = 0$. Basically, the $\phi_{k,l}$ phases are uniformly distributed on the range $[0, 2\pi)$, as e.g. in the Rayleigh-model [113].

While the cluster arrival times make up a Poisson process, for simulating the arrival times, only the first cluster needs to be accounted – the effects of late components can be neglected since cluster arrival rate is much lower than the component arrival rate –, so in the following, only one cluster is assumed (i.e. $L = 1$). The original SV model also modeled the component arrival times as a Poisson process, but the IEEE model uses a mixture of two Poisson processes, i.e.

$$p(\tau_{k,l}|\tau_{k-1,l}) = \beta\lambda_1 e^{-\lambda_1(\tau_{k,l}-\tau_{k-1,l})} + (1-\beta)\lambda_2 e^{-\lambda_2(\tau_{k,l}-\tau_{k-1,l})} \quad (5.2)$$

Here, β is the mixture probability, λ_1 and λ_2 are the ray arrival rates. The mixture distribution is especially useful for modeling the diversity of ray arrival times in indoor environments.

The **power delay profile (PDP)** is an important part of the model as it determines the main statistical parameters for the small-scale fading process. The model specifies two PDPs, one for the original model and one for some NLOS cases in indoor situations (especially in office spaces). The original PDP is exponential:

$$\mathbb{E}[|a_{k,l}|^2] = \Omega_l \frac{1}{\gamma_l((1-\beta)\lambda_1 + \beta\lambda_2 + 1)} e^{-\tau_{k,l}/\gamma_l} \quad (5.3)$$

, where Ω_l is the energy of the PDP of the l th cluster, and γ_l is the decay of the PDP. For indoor NLOS cases, it is observed that the power of the first cluster is increasing first, reaches a maxima, then decreases again. For such cases, the proposed PDP is [114]:

$$\mathbb{E}[|a_{k,0}|^2] = (1 - \chi e^{-\tau_{k,0}/\gamma_{\text{rise}}}) \cdot e^{\tau_{k,0}/\gamma} \frac{\gamma + \gamma_{\text{rise}}}{\gamma} \frac{\Omega_0}{\gamma + \gamma_{\text{rise}}(1 - \chi)} \quad (5.4)$$

Here, χ is the attenuation of the first component, γ_{rise} describes the increased speed of the PDP, while γ describes the decay. Figure 5.1 shows the typical form of the PDPs shown in (5.3) and (5.4).

The decay of the clusters, i.e., γ_l , depends on the cluster index linearly. However, the first cluster decay equals γ_0 , which is specified a priori. Also, the model states that Ω_l varies in cluster index, but Ω_0 is log normally distributed [111], i.e.

$$10 \log \Omega_0 = M_{\text{cl}} \quad (5.5)$$

where M_{cl} is normally distributed, $M_{\text{cl}} \sim \mathcal{N}(0, \sigma_{\text{cl}}^2)$. Finally, small-scale fading is controlled by a Nakagami distribution [115]:

$$p(|a_{k,l}|; m_{k,l}, \Omega_{k,l}) = \frac{2m_{k,l}^{m_{k,l}}}{\Gamma(m_{k,l})\Omega_{k,l}^{m_{k,l}}} |a_{k,l}|^{2m_{k,l}-1} \exp\left(-\frac{m_{k,l}}{\Omega_{k,l}} |a_{k,l}|^2\right) \quad (5.6)$$

where $\Gamma(\cdot)$ is the Gamma-function. The $\Omega_{k,l}$ parameter corresponds to the mean power of the component, so it can be computed from the PDP. The $m_{k,l}$ parameter is modeled as a log-normal random variable, with μ_m and σ_m^2 specified a priori.

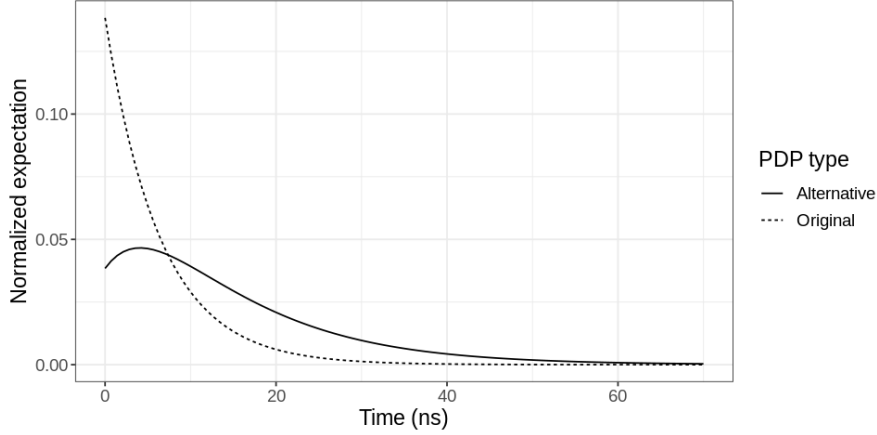


Figure 5.1: Typical normalized power density profiles used in the simulations. The original PDP is basically exponential, while the alternative PDP for NLOS cases has an increment before the decay.

5.1.2 Pulse shaping

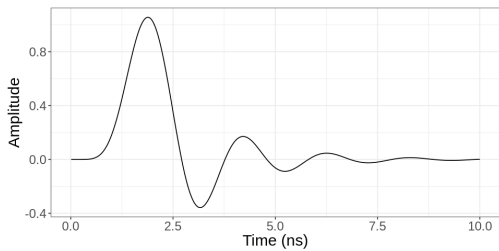
Based on the channel impulse response, the received signal can be simulated by convolving the channel impulse response with the impulse response of the pulse shaping filter, i.e.

$$h_r(t) = h(t) * g(t) \quad (5.7)$$

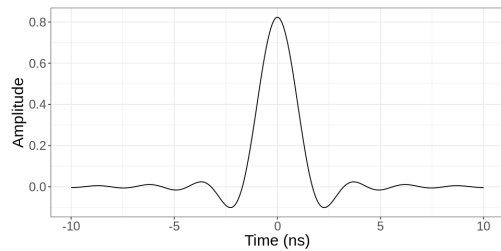
In the IEEE 802.15.4a standard, there are a couple of restrictions and requirements regarding the pulse used by the transmitter. However, the reference pulse is the impulse response of a root-raised cosine filter. For example, the pulse duration for channel 5 is $T_p = 2.0$ ns, and the roll-off factor is $\beta = 0.6$. In real systems, the non-causality of the raised cosine pulse and the complex implementation of the filter give way to other pulse shapes, as well. The standard recommends the impulse response of an 8 degree Butterworth filter, which is a causal pulse with a short pulse duration. Investigating the actual pulse shape of e.g. the Qorvo DW1000 chip, it is also proved to be some asymmetric pulse [67]. Figure 5.2 shows the plots of the typical reference and Butterworth pulses for channel 5 in UWB systems.

5.1.3 Noise

For simulating the noise at the receiver, a band-limited Gaussian noise can be added to the generated signal with N_0 power spectral density. Note that very high noise power (i.e., low signal-noise ratio) may affect the distribution of the receive timestamps because the Gaussian distribution may dominate the timestamping process. Different values of noise power shall be investigated in the simulation to examine this situation.



(a) Impulse response of an eight order low-pass Butterworth filter, with a bandwidth of 500 MHz



(b) The reference root raised cosine pulse for channel five ($T_p = 2$ ns, $\beta = 0.6$)

Figure 5.2: Typical pulse shapes in UWB systems. The pulses are normalized to unit energy.

5.1.4 Timestamping

While the actual timestamping process can be done by a variety of methods, the most basic one defines a threshold and searches for the first pulse in the received signal crossing this threshold. However, the exact methods of determining the threshold value are often confidential, but mostly, they are based on the estimated noise level. Also, interpolation between the samples of the impulse response is typically done to get a more precise estimation of the time-of-arrival of the signal rather than the time resolution of the digital system. For example, the Quorvo DW1000 chip applies a [Leading Edge Detector \(LED\)](#) module, which uses the same method for extracting the RX timestamp [116].

Figure 5.3 shows a sample from the channel response presenting the receiver threshold and also the instant of the estimated reception.

5.1.5 Results

The simulations are based on the final report of the IEEE 802.15.4a channel model [111], where a reference implementation of the channel model can be found using MATLAB scripts. However, in this work, the simulations were performed using custom-developed scripts written in the R language by the author. In each simulation 2000 received samples were generated with at most 200 ray components. Both the original PDP (see (5.3)) and the alternative PDP (see (5.4)) were evaluated.

The parameters of the model used in the process can be found in Table 5.1., which are the recommended values in dense indoor office environments [111]. The timestamping was performed at the level of 0.1 amplitude, which is somehow arbitrary; however, changing the level gives essentially identical results. The pulse shapes were normalized to unit energy. The noise level in the simulation is assumed to be $N_0 = -137$ dBm/Hz. The simulation was performed with a resolution of $1/F_s = 10$ ps, where F_s is the sampling frequency. For each scenario, 2000 channel impulse response samples were drawn from the model. A couple of random samples are plotted in Figure 5.4, where the alternative PDP was simulated with Butterworth filtering.

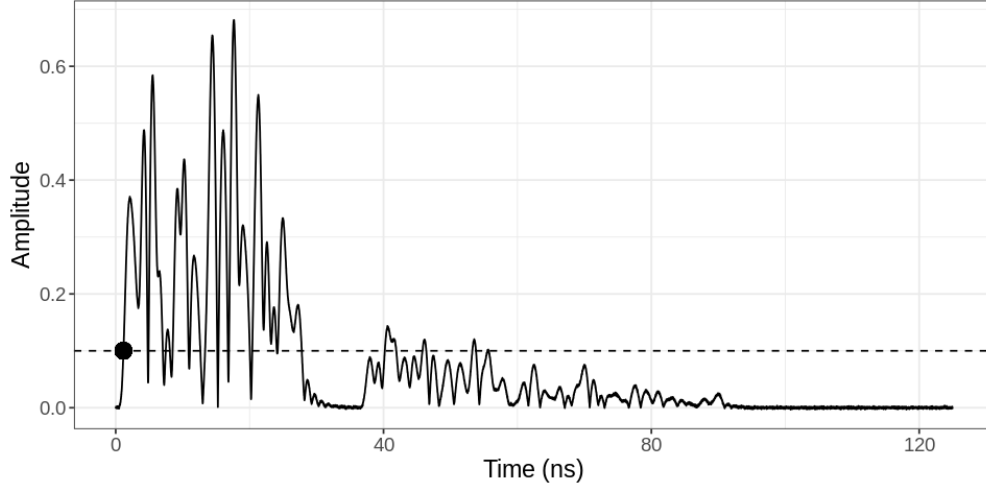


Figure 5.3: A sample of the channel response showing the receiver threshold and the instant of the timestamping (black dot).

Table 5.1: The channel model parameters used for simulation.

Parameter	Value
$\lambda_1, \lambda_2, \beta$	$0.11 \frac{1}{\text{ns}}, 2.09 \frac{1}{\text{ns}}, 0.0096$
γ_0	6.4 ns
σ_{cl}	3 dB
γ	11.84
γ_{rise}	11.21
χ	0.7
μ_m	0.5 dB
σ_m	0.25 dB

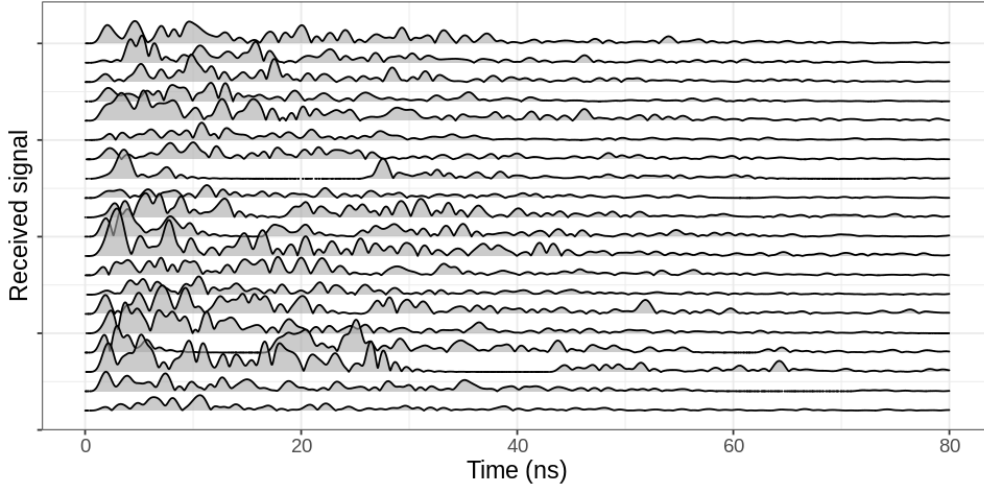


Figure 5.4: Random samples from the simulated received signals. The samples are taken from the distribution of the alternative PDP using a Butterworth filter. The samples contain no noise.

Figure 5.5 shows the result of the simulations. Three scenarios were investigated: the alternative PDP (see (5.4)) with **root raised cosine (RRC)** pulse and with Butterworth pulse, and the original PDP (see (5.3)) with Butterworth pulse. For the RRC pulse, some delay compensation was required because of the filter’s non-causality. As can be seen, the distributions are asymmetric and skewed, not like the Gaussian distribution. For each scenario, the distributions seem similar, but the parameters are somewhat different.

Figure 5.7 shows the noise power dependency of the distribution of the receive timestamp. In the figure, different levels of noise were applied to the received signal to observe the perturbation of the distribution. However, the shape and the quality of the distribution seem to be – except for some extreme values – invariant to the noise power.

For the skewed distribution, three well-known candidate distributions were fit to the empirical data using maximum likelihood techniques. These distributions were selected because they can be described in closed form, they are skewed distributions, and it is easy to find conjugate priors for them. Figure 5.6. shows the inverse gamma, the gamma, and the log-normal distribution fit to the samples along with the reference Gaussian distribution. It is readily seen that the Gaussian distribution is just too general a model for the distribution of the timestamps. The other three – especially the log-normal distribution – fit well to the distribution.

Table 5.2 shows the log-likelihood values for each model in each scenario. The log-likelihood can be seen as the evidence function for the probability densities, helping us to select the proper distribution describing the samples. In all three scenarios, the log-normal distribution shows the best fit for the data. Thus, it can be recommended to use it as the distribution of the receive timestamp instead of the Gaussian distribution. Using the proper distribution for any specific inference problem based on the received

Table 5.2: The log-likelihood (evidence) of the selected distributions in the simulated scenarios (higher is better). In the bold typeset, the maximum value is represented for the scenarios. In each scenario, the log-normal distribution has the biggest likelihood value.

	Normal	Log normal	Inverse gamma	Gamma
RRC alternative PDP	-11,291.14	-10,468.01	-10,652.64	-10,565.19
Butter alt. PDP	-10,721.37	-9,658.15	-9,708.85	-9,858.44
Butter orig. PDP	-9,344.64	-8,658.89	-8,763.65	-8,751.14

timestamp, e.g., for synchronization or localization problems, results in more accurate time or location estimation.

5.2 Measuring the distribution of receive timestamps in UWB systems

To measure the real distribution of the receive timestamps in UWB systems, we have to synchronize the devices involved in the measurement. However, while measuring the frequency drift between devices is viable, it is hard to synchronize the measurement with the message transmission procedure. Suppose we have two devices, the first one running on wall-clock time and sending its first message at t_1 . The second one measures the time from some offset o , but with the same frequency. Thus, the measurements can be expressed in the second device as

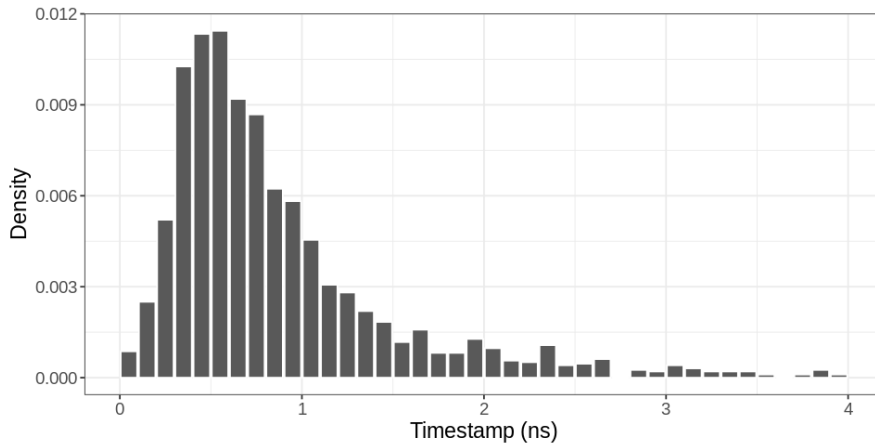
$$y_k = (t_k - t_1) + \tau_{\text{ToF}} + o + \omega_k \quad (5.8)$$

where o is a constant offset, ω_k is the phase noise at step k , τ_{ToF} is the propagation delay, and y_k is the measurement at step k . The quantity

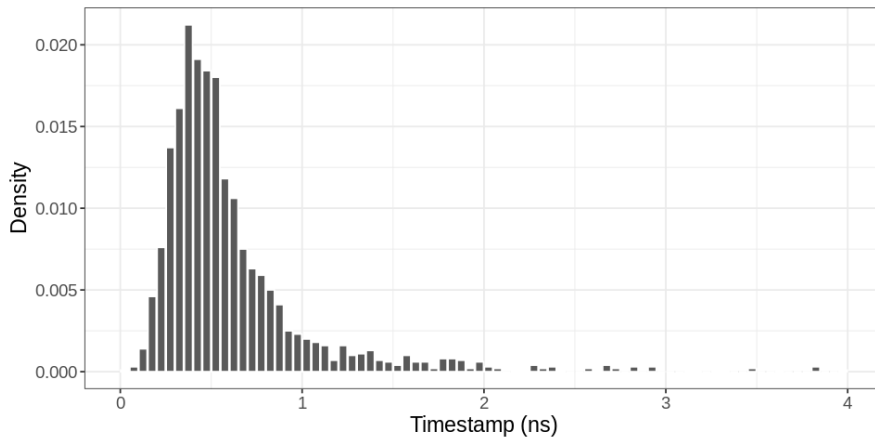
$$y_k - t_k = \omega_k + (-t_1 + \tau_{\text{ToF}} + o) = \omega_k + \text{const.} \quad (5.9)$$

consists of the random variable ω_k and a constant if τ_{ToF} is held constant, e.g., placing the devices to fix locations or keeping their distance constant. This way, the measurement noise can be measured *up to a constant*, considering that the frequency of the oscillators on the devices is synchronized (i.e. synthonized).

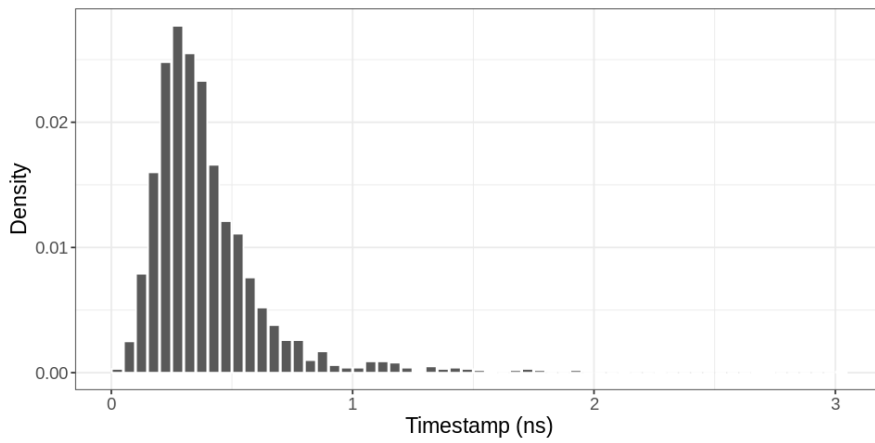
Figure 5.8 shows the layout of the measurement. The synchronization is implemented between two Decawave DWM1001 modules, which are capable of sending and receiving ultra wide band signals, and precisely timestamping the transmission and reception of messages on a clock running at cca. 64GHz. To ensure $f(t) = 1$, the two modules are driven from a common **TCXO** oscillator. Since the clock signal is transmitted through a long coaxial cable (typically a couple of meters), a clock buffer shall be used. However, as the DW1000 chip on the DWM1001 module requires a clock signal with a very small



(a) Root raised cosine filter, alternative PDP



(b) Butterworth filter, alternative PDP



(c) Butterworth filter, original PDP

Figure 5.5: Distribution of the receive timestamps based on the simulation of 2000 samples in each scenario.

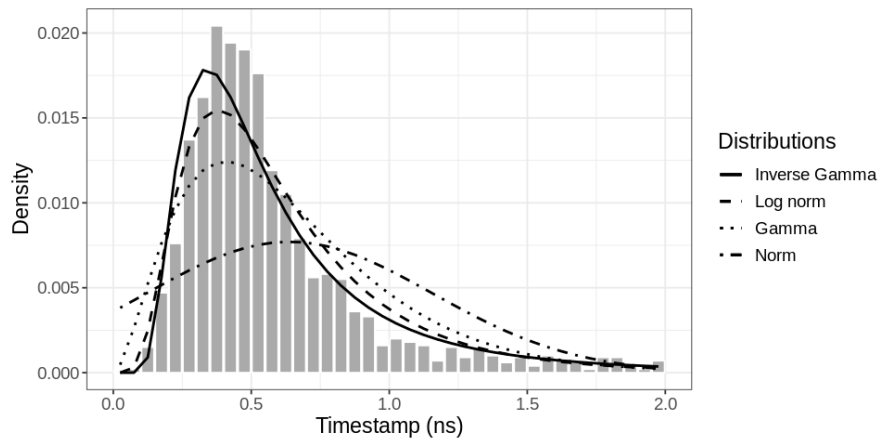


Figure 5.6: Maximum likelihood estimates of different asymmetric probability distributions versus the empirical probability density in the case of the Butterworth pulse for the alternative PDP. It is readily visible that the distribution is far away from a Gaussian distribution.

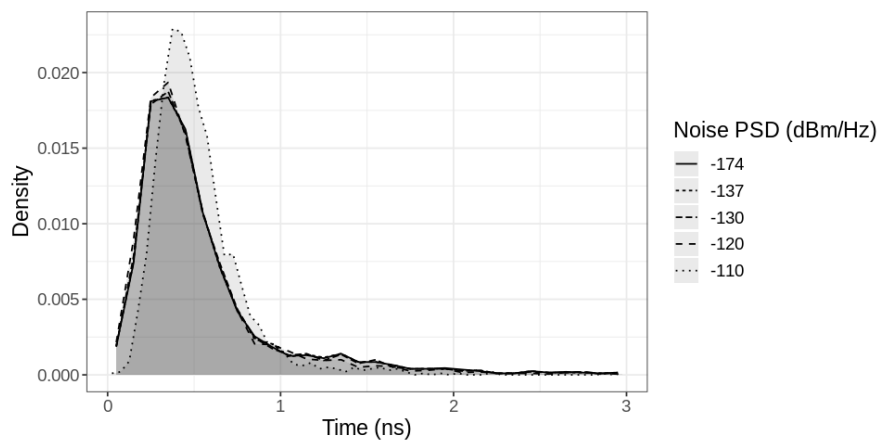


Figure 5.7: Noise dependency of the distribution of receiving timestamps. The noise values range from the thermal noise to as extreme as -110 dBm/Hz. The shape and the quality of the distribution do not change radically when noise power is increased.

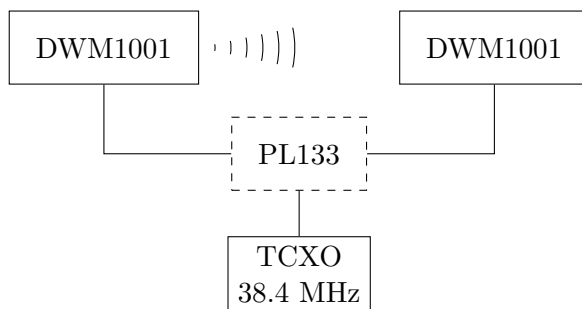


Figure 5.8: The measurement layout for the synchronized measurements. The reference clock of the thermal controlled oscillator is distributed by an extremely low additive phase noise PL133 clock distributor IC.

phase noise (e.g. -132 dBc/Hz@1 kHz), an appropriate PL133 clock distributor IC is in operation between the devices.

Note that while driving the clocks from the same TCXO oscillator, most of the bias is ceased; however, phase jitter at the transmitters can be different, resulting from the propagation characteristics on the wire. In the installation, the two clock wires have the same length to ensure that phase variations at the oscillator propagate to the transmitter at the same time. The only clock discrepancy left between the transmitters is caused by small wire capacity deviations and the analog circuits of the DW1000 chip (e.g., the PLL filter, etc.), which can be neglected compared to the precision of the algorithm.

To validate the presented method, ultra wide band measurements were made in three different scenarios. Two devices were located at a distance of 4.5 meters, and one of the devices periodically sent a message to the other device, recording its transmit timestamp. The message transmission period was approximately 50 milliseconds.

1. *Calm environment*: in the evening, when little or no movement can be experienced
2. *Normal environment*: a normal workday, medium activity in the building
3. *Harsh environment*: lots of movements, high activity

The distribution of the ω_k receive timestamp errors (up to a constant) can be seen in Figure 5.9 for all three scenarios. Note that all the distributions are somewhat multimodal. However, the distribution in the harsh indoor environment has two very distinct modes. It can be readily seen in the figures that the deviation of the measurement noise increases with the activity in the building.

To get some insight into the timestamping process, Figure 5.10 shows real measurements of UWB channel impulse responses, 50 samples each. The CIRs are plotted relative to each other by the timing of the radio receiver synchronization circuit, not to some absolute time frame. The figures also present the detected first path samples by red lines at each CIR sample. Figure 5.10a shows the situation of the multimodal timestamp distribution, where it can be clearly seen that the first component is so weak

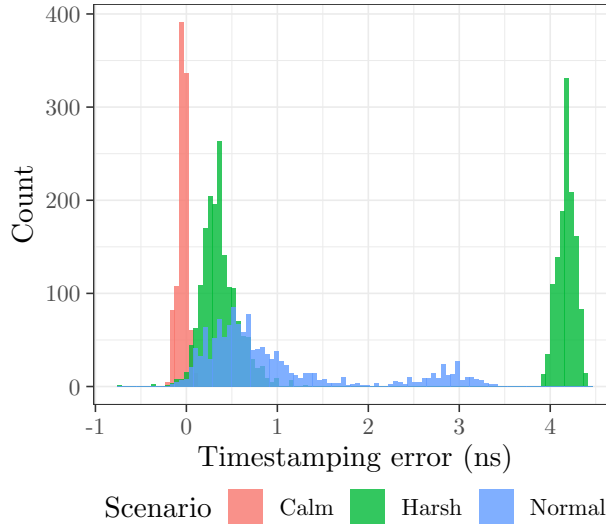
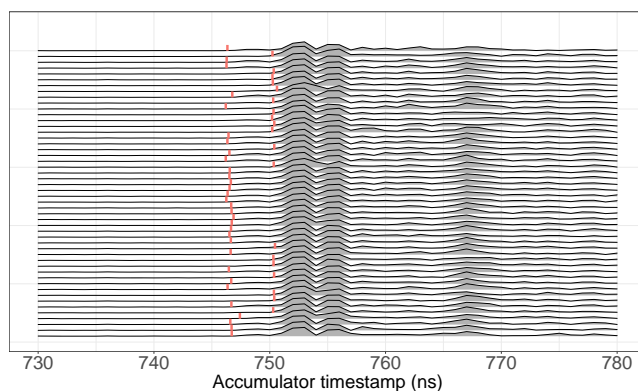


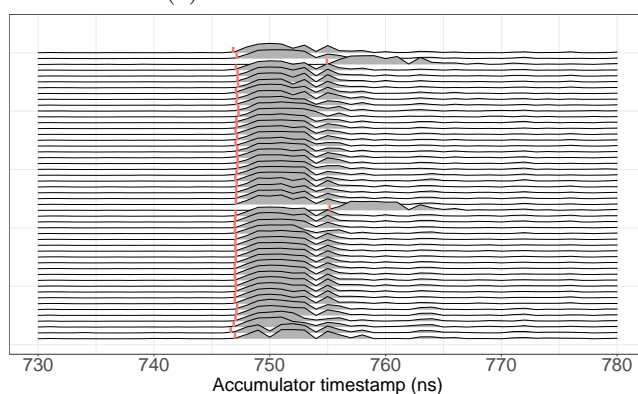
Figure 5.9: Distribution of the ω_k measurement noise in three different measurement scenarios. The timestamping error is specified up to a constant, as in (5.9). Note that the distribution in a harsh environment is strongly multimodal.

that it sometimes hits the threshold of the LED circuit sometimes it hides into the noise. Figure 5.10b presents a calm situation where the UWB devices are close to each other (32 cm). Note that this case shows a narrow timestamping distribution. Also, the receiver sometimes synchronizes to a different part of the radio signal, resulting in a slip in the CIR plot. Figure 5.10c shows a common situation when the receiver is moving randomly. Please note that the CIRs are very different in each moment, and the timestamping algorithm (i.e., LED) has a hard time estimating the first path, resulting in uncertain first-path estimations.

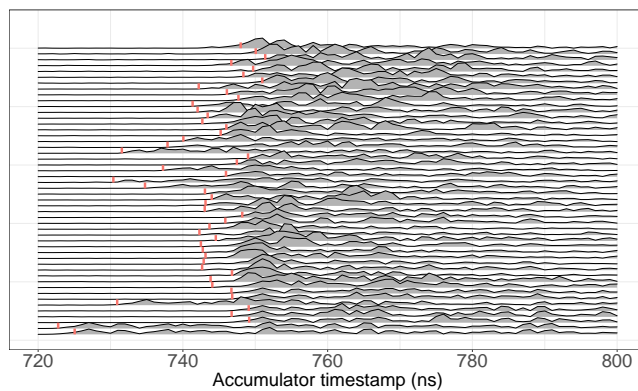
The primary conclusion from the measurements is that the distributions of the received timestamps across all scenarios are skewed, exhibit a wide range of error variances, and are sometimes multimodal. Multimodality presents a new challenge for inference algorithms: when dealing with multimodal distributions, the estimated quantities can exhibit more modes than the original measurements, especially as inference is conducted over longer periods. However, this work focuses on addressing the wide range of variances—specifically, how to estimate the variance of the received timestamps during inference. By doing so, the multiplicative effect of multimodality can be mitigated. Note that this requires modeling the measurement noise as a Gaussian random variable – resulting in a suboptimal estimation since we are using a rather general distribution. However, estimating the measurement noise variance can substantially improve inference accuracy, which will be presented in the next.



(a) Multimodal detection



(b) Calm and near communication



(c) Moving device

Figure 5.10: Ultra wide band channel impulse response plots recorded by real measurements to perceive the real behavior of the wide-band wireless channel. The plots show the magnitude (i.e., absolute value) of the channel impulse response (without units only for illustrative purposes). The X-axis shows the time measured by the sample accumulator, while the Y-axis shows the sample amplitude. 50 signals are presented as ridge lines in each scenario. The red lines show the first path arrival recognized by the DW1000 leading edge detector. The samples are fitted to each other by the radio synchronization circuit, so limits of the X axis are selected to show relevant parts.

5.3 Estimating measurement noise of the distribution of the received timestamps

As demonstrated in the preceding sections, the wide-ranging distribution of receive timestamps (especially the wide range of variances) poses a challenge to achieving low bias and variance in the time estimation process. At the core of every time synchronization system lie clock synchronization algorithms, tasked with offering offset and frequency drift estimations derived from time measurements affected by noise. A couple of efficient algorithms are developed over time, e.g., window-based solutions [55], probability filtering solutions, like Kalman-filters [62, 117] or nonlinear, variational Bayes adaptive filtering [118]. Seijo et al. [1] shows a method for IEEE 802.11 systems to provide precise timestamping in the nanosecond range. However, it requires the channel impulse response samples to be available. While Kalman-filter-based solutions give decent results, the algorithms are sensitive to the estimated measurement noise variance [62], which is typically unknown and estimated or measured a priori. For a couple of application areas, the measurement noise is shown to be not even Gaussian, e.g. in dense indoor UWB communication, the distribution of receive timestamps can be log-normal (see Section 5.1).

The estimation of measurement noise can substantially improve the accuracy of the state estimation. In the following, a fully Bayesian method of frequency drift estimation will be introduced, which can be applied to estimate not only the frequency drift itself but also the optimal variance of the measurement noise. It is shown that using the optimal measurement noise variance, the frequency drift estimation gives better results, especially in the case of highly uncertain measurements. Subsection 5.3.4. evaluates the algorithm by estimating the frequency drift of two ultra wide band devices, where the measurement noise is mainly caused by multipath propagation.

5.3.1 Measurement method

For estimating the frequency drift between two devices, the basic, implementation agnostic measurement scheme is depicted in Figure 5.11. Sometimes, real implementations have additive messages transmitted since transmission timestamps cannot be included in the message (e.g., in the case of IEEE 1588 systems, SYNC and FOLLOW-UP messages, see Figure 2.6), but in theory, the transmission and reception timestamps are recorded and used for synchronization. It is worth mentioning that PTP uses the SYNC messages for synthonization in transparent clocks. However, for the moment, it is sufficient to use the model in Figure 5.11, which is a general form for stating the mathematical model. Since the real frequency of either clock cannot be estimated without proper reference (e.g., atomic clock), the master clock is supposed to be the reference clock to which the slave clock wants to synchronize its frequency drift. Please note that the master clock in this context is not a grandmaster clock, but the clock to which the slave clock synchronizes in a master-slave situation.

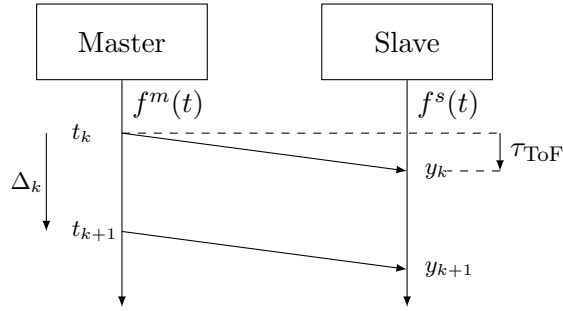


Figure 5.11: Synchronization messages from the master clock to the slave clock. The reference clock is the master clock, for which the time instants are denoted as t_k , while slave clock measurements are denoted as y_k . Time-of-flight is supposed to be constant.

Let $\psi(t) = f^s(t)/f^m(t)$ denote the time-dependent relative frequency of the slave clock (where $f^m(t)$ and $f^s(t)$ are the time-dependent frequency of the master and slave clock respectively); the measurements can be written as

$$y_k = l(t_k) + \omega_k$$

$$l(t_k) \equiv l_k = \int_0^{t_k} \psi(t) dt + \tau_{\text{ToF}} + o \quad (5.10)$$

, where y_k is the value of the slave clock at the reception of the synchronization message for the message transmitted at t_k with $k = 1, 2, 3, \dots$, τ_{ToF} is the time-of-flight, o is a constant, time-independent offset, and ω_k is the measurement noise with probability density $p(\omega_k)$. Note that the measurement noise represents the uncertainty of the time-stamping process, not the propagation noise on the wireless channel. The resolution of the digital clocks are neglected as they are supposed to be high enough (e.g. the DW1000 chip has 15ps resolution). $l(t_k)$ or simply l_k is the noise-free measurement of the clock value at time instant t_k .

The main task is now to estimate the values of $\psi(t)$ at time instants t_k denoted by $\psi_k \equiv \psi(t_k)$ from the measurements y_k . Here, some simplifications are carried out. The time-of-flight is assumed to be constant, i.e., the distance of the devices is constant and time-independent. It is assumed also that $\psi(t_k) \approx \psi(t_k + \tau_{\text{ToF}})$, i.e., the change of the frequency drift can be neglected under the time-of-flight. Also, o is assumed to be zero (i.e. $o = 0$).

From a Bayesian point of view, we want to estimate the joint probability density of $\psi_{1:k} = \{\psi_1, \psi_2, \dots, \psi_k\}$ conditioned on the measurements $\mathbf{y}_{1:k} = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_k\}$, i.e. the probability

$$p(\psi_{1:k} | \mathbf{y}_{1:k}) \quad (5.11)$$

However, estimating the joint probability is hard and generally intractable. It is also expected that the algorithm estimating the frequency drift could be calculated online, receiving measurement y_k one-by-one. The solution for such problems is the

well-known *linear dynamic system*, providing efficient tools for estimating latent states to measurements.

5.3.2 Linear filter model

The linear dynamic system used in frequency drift estimation can be described by a first-order Markov process, expressing the $p(\mathbf{x}_{k+1}|\mathbf{x}_k)$ state transition and $p(\mathbf{y}_k|\mathbf{x}_k)$ measurement probabilities, where \mathbf{x}_k is the state of the linear filter at time instant t_k .

Dynamic model

For the dynamic model, the method uses the state space model presented in Section 2.2.3, with A_k state transition matrix and Σ_k state transition covariance matrix (i.e. process dynamic noise covariance).

Measurement model

The latent states, i.e., the clock dynamics, result in measurements controlled by the measurement model. The measurement model is a simple linear model, in which the clock value is measured directly accompanied by a specific measurement noise:

$$p(\mathbf{y}_k|\mathbf{x}_k) = \mathcal{N}(\mathbf{y}_k|H\mathbf{x}_k, \xi) \quad (5.12)$$

where

$$H = [1 \quad 0 \quad 0] \quad (5.13)$$

Here, the measurements are directly the clock readings since time-of-flight is neglected (see 5.3.1).

Estimating the states

Assuming a known ξ measurement noise variance, the estimation of the \mathbf{x}_k states at time instant t_k in a first-order Markov process can be done by using the well-known recursive Kalman-filter equations (see (2.5) and (2.6)). Kalman-filter equations estimates the

$$p(\mathbf{x}_k|\mathbf{y}_{1:k}, \xi) \quad (5.14)$$

filter density recursively, starting from a prior distribution

$$p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1}, \xi) \quad (5.15)$$

Note that the dependence on the – presumably – unknown and unobserved ξ measurement variance is made explicit. The Bayesian estimation of the filter density is $p(\mathbf{x}_k|\mathbf{y}_{1:k}, \xi) = \mathcal{N}(\mathbf{m}_k, P_k)$, where \mathbf{m}_k and P_k can be calculated using the Kalman-filtering equations in (2.5) and (2.6). The Kalman-filtering equations are optimal in the sense of maximum likelihood estimation, given the known ξ variance. The K_k Kalman-gain is the optimal interpolator only if the variances agree with the variances of the true distributions, which are – as in most cases – unknown and shall be estimated.

5.3.3 Estimating the measurement noise

For a first-order Markov chain, the assumption of Gaussian dynamic and measurement noise makes it possible to easily derive the posterior (or the filter) distribution of the latent states. However, measurement noise is not strictly normal; moreover, in harsh indoor environments, the distribution of the measurement noise can even be multimodal. Multimodal noise is hard to handle: e.g., the Bayesian filter for the first-order Markov chain can result in multimodal marginal state distribution, which recursively adds more and more mode to the state distribution.

To keep the complexity low, the Gaussian distribution assumption is kept; however, to head toward the optimum estimation, the ξ measurement noise estimation is necessary. Suppose K measurements, and $k = 1, 2, \dots, K$. From a Bayesian perspective, estimating the ξ covariance equals the evaluation of the probability density $p(\xi|\mathbf{y}_{1:K})$. Using maximum a posteriori (MAP) estimation, the optimal estimation of the ξ covariance is

$$\xi_{\text{MAP}} = \arg \max_{\xi} p(\xi|\mathbf{y}_{1:K}) \quad (5.16)$$

The direct maximization of the (5.16) results in the maximization of the integral

$$p(\xi|\mathbf{y}_{1:K}) = \int p(\mathbf{x}_{0:K}, \xi|\mathbf{y}_{1:K}) d\mathbf{x}_{0:K} \quad (5.17)$$

for the whole state space, which is computationally hard to evaluate, and it grows exponentially with increasing K . (Note that in discrete state spaces, the integral becomes summation.)

However, for these kinds of latent space problems, *Expectation-Maximization* (EM) method can help to estimate the distribution of the parameter by estimating the distribution of the parameter and the latent space separately [119]. Expectation maximization in the context of parameter estimation in latent spaces results in computing the expectation

$$Q(\xi, \xi^{(n)}) = \mathbb{E}_{\mathbf{x}_{0:K}|\mathbf{y}_{1:K}, \xi^{(n)}} [\log p(\mathbf{x}_{0:K}, \mathbf{y}_{1:K}|\xi)] \quad (5.18)$$

where $Q(\xi, \xi^{(n)})$ is a function depending on ξ . In the next step, we need to maximize it for the next iteration:

$$\xi^{(n+1)} = \arg \max_{\xi} Q(\xi, \xi^{(n)}) \quad (5.19)$$

Note that the calculation of the expectation is much easier since it requires estimating the joint distribution of the latent states given a measurement noise and then maximizing the log-likelihood knowing the latent states. The maximization in (5.19) has proven to increase the likelihood value and has proven to converge at least to a local maxima [119].

The maximization can be achieved by various methods, e.g., gradient-based method, like [Stochastic Gradient Descent \(SGD\)](#) or the [Broyden–Fletcher–Goldfarb–Shanno algorithm \(BFGS\)](#) [120]. However, for this convex quadratic problem, the maxima can be calculated in a closed form. Using (5.12), the log-likelihood of the measurement

factorization (2.3b) – assuming an irregular, noninformative prior – yields

$$\ln \mathcal{L}(\xi; \mathbf{x}_{0:K}, \mathbf{y}_{1:K}) = -\frac{K}{2} \ln |\xi| - \frac{1}{2} \sum_{k=1}^K (\mathbf{y}_k - H\mathbf{x}_k)^T \xi^{-1} (\mathbf{y}_k - H\mathbf{x}_k) + \text{const.} \quad (5.20)$$

where all terms independent of ξ are collected into a single constant. Now, the maxima of the likelihood can be acquired by setting the derivative to zero, which turns out to be the maximum likelihood estimate of the covariance of a Gaussian random variable:

$$\xi_{ML} = \frac{1}{K} \sum_{k=1}^K (\mathbf{y}_k - H\mathbf{x}_k)(\mathbf{y}_k - H\mathbf{x}_k)^T \quad (5.21)$$

Note, that the estimate of the measurement variance ξ can intuitively be interpreted as the variance of the error between the real measurements and the predicted measurements based on the estimated latent states.

For calculating (5.18), $p(\mathbf{x}_k | \mathbf{y}_{1:K})$ needs to be expressed, however, Kalman-filter provides only $p(\mathbf{x}_k | \mathbf{y}_{1:k})$, i.e. the optimal estimation of the latent states shall be carried out, and the future observations need to be accounted, as well. For a first-order Markov process, the factorization (2.3a) and (2.3b) makes it possible to use the [Rauch-Tung-Striebel smoother \(RTS\)](#) in (2.8).

The summary of the algorithm can be found in Algorithm 1 for a W wide window of measurements. The algorithm estimates the measurement noise variance and the clock state at timestep k based on the last W measurements. The algorithm basically tries to enhance the estimation made by a standard Kalman filter (using the estimated measurement covariance), as presented in the first step. After having the estimated state (i.e. frequency), N_{EM} iteration of covariance estimate refinement is done. First, an RTS smoother is applied to the last W measurement, from which the measurement noise is estimated by using (5.21). Using the refined value of ξ_k , the W wide window of states is reestimated by applying Kalman-filtering.

5.3.4 Results

For the evaluation, measurements presented in Section 5.2 are applied. However, using measurements with $\psi(t) = 1$ solely (remember, $\psi(t)$ is the time-dependent relative frequency of the slave clock), the bias following the capability of the algorithms cannot be observed since the optimal solution in estimating the frequency drift process shall satisfy the bias-variance compromise. E.g., using a low-pass filter where the f_c cut-off frequency tends to $f_c \rightarrow 0$, the variance will tend to zero. However, this low-pass filter will be unable to follow the bias. To evaluate the algorithms with biased measurements, artificial bias is added to the frequency drift while processing based on the model in (2.15).

The estimation was performed using the equations presented in Algorithm 1. For the dynamic noise and the measurement noise, the values in [121] are used, where $\sqrt{\xi} = 0.13\text{ns}$ and $\sigma_3 = 51 \cdot 10^{-9}$, while $\sigma_1 = \sigma_2 = 0$ and the same value is used for

Algorithm 1 Step for the joint estimation of the clock state \mathbf{x}_k and the measurement noise variance ξ_k at timestep k . $\text{KF}(\cdot)$ represents Kalman-filtering, while $\text{RTS}(\cdot)$ represents Rauch-Tung-Striebel smoothing.

Input:

- k : Actual timestep index
- N_{EM} : Expectation-Maximization iteration count
- W : Window size
- $A_{(k-W):k}$: Last W dynamic matrices
- $H_{(k-W):k}$: Last W measurement matrices
- $\mathbf{x}_{(k-W):(k-1)}$: Last $W - 1$ state means
- $P_{(k-W):(k-1)}$: Last $W - 1$ covariance matrices
- $\sigma_1^2, \sigma_2^2, \sigma_3^2, \xi_k$: Actual dynamic and measurement noise variances
- $\mathbf{y}_{(k-W):k}$: Last W measurements

Output:

- \mathbf{x}_k, P_k : Actual state
- ξ_k : Estimated measurement noise variance

Procedure

- 1: $\mathbf{x}_k, P_k \leftarrow \text{KF}(\mathbf{x}_{k-1}, P_{k-1}, A_k, H_k, \sigma_1^2, \sigma_2^2, \sigma_3^2, \xi_k)$
 - 2: **for** $i \leftarrow 1$ **to** N_{EM} **do**
 - 3: $\mathbf{x}_{(k-W):k}^{RTS} \leftarrow \text{RTS}(W, \mathbf{x}_{(k-W):k}, P_{(k-W):k})$
 - 4: $\xi_k \leftarrow \frac{1}{W} \sum_{i=k-W}^k (\mathbf{y}_i - H_i \mathbf{x}_i^{RTS})^2$
 - 5: $\mathbf{x}_{(k-W):k}, P_{(k-W):k} \leftarrow \text{KF}(\mathbf{x}_{k-W}, P_{k-W}, A_{(k-W):k}, H_{(k-W):k}, \sigma_1^2, \sigma_2^2, \sigma_3^2, \xi_k)$
 - 6: **end for**
 - 7: **return** \mathbf{x}_k, P_k, ξ_k
-

the expectation-maximization algorithm as the initial value for ξ . Three results are presented:

1. The original Kalman filter-based solution, as in [121]. The values for the dynamic and the measurement noises are the same as above. (KF)
2. The Kalman filter, with the estimated measurement noise. The dynamic noise is the same as above. (KF-NE, NE stands for „Noise Estimation”)
3. The Rauch–Tung–Striebel (RTS) smoother, with the estimated measurement noise. The dynamic noise is the same as above. (RTS-NE)

Note that the RTS smoother is unable to work as a filter, even in the case of the assumption that the measurement noise is stationary, since the RTS smoother needs the measurement samples from the future, as well. So, comparing RTS smoother to Kalman-filtering is unfair; however, the results are presented to represent the optimal joint solution.

For the harsh scenario, the measurement noise standard deviation resulted in the iterations of the expectation-maximization method, which is presented in Figure 5.12. Starting from the initial value in step 0, the convergence is really fast, and the final value is reached in one or two iterations.

Figure 5.13 shows an example of the estimated frequency drifts by the three algorithms along with the ground-truth frequency drift. Note the bias of the frequency drift in time: the bias is introduced artificially using the dynamic model (2.15) to evaluate the bias following the capability of the algorithms.

The main metric for comparing the solutions is the standard deviation of the frequency drift error. Frequency drift error is defined as $\epsilon = \dot{l}(t_k) - (1 + \dot{l}_{\text{bias}}(t_k))$, where $\dot{l}(t)$ is the time derivative of $l(t)$. Since the devices in the synchronization process have the same frequency, i.e., $f^s(t) = f^m(t)$, 1 stands for the identical frequency. Also, $\dot{l}_{\text{bias}}(t_k)$ stands for the artificially introduced bias, which is known ahead as a ground-truth bias. The standard deviation of the drift error can be seen, as the root mean squared error.

Figure 5.14 shows the error of the estimated frequency drift error in time for the three scenarios. Note that in harsh indoor environments, the original Kalman filter solution acts much worse than the optimal Kalman filter solution or the RTS smoother. In all scenarios, the RTS smoother gives the best results; however, the smoother cannot act as a filter, so it is unable to act as a real-time solution. Note also, that no visible bias can be recognized in the error plots.

Figure 5.15 shows the absolute frequency drift errors as empirical cumulative distribution functions. The same conclusion can be drawn that the Kalman filter with the estimated measurement noise acts better than the original Kalman filter solution in all scenarios. Note that in the case of the calm and normal scenario, the difference between the optimal Kalman filter and the original one is very small; however, using the optimal measurement noise estimate results in fewer overshoots (i.e., smaller standard deviation).

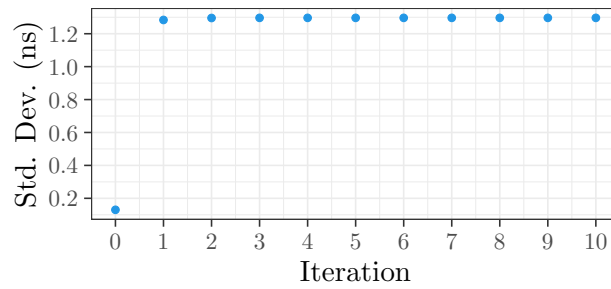


Figure 5.12: Measurement noise estimation against expectation-maximization iteration. The convergence to the optimal measurement noise estimate is achieved in as few as one or two iterations. Note that step 0 is the original value of the standard deviation before refinement.

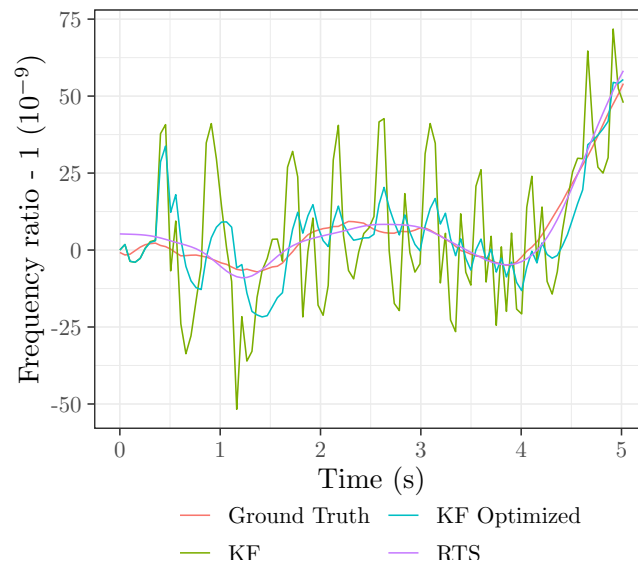
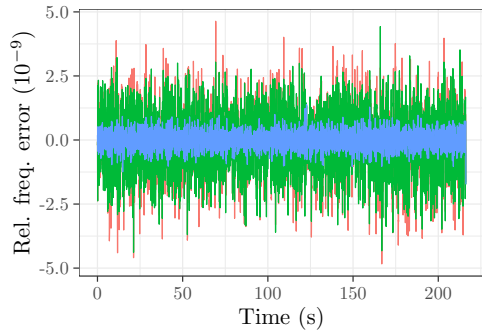
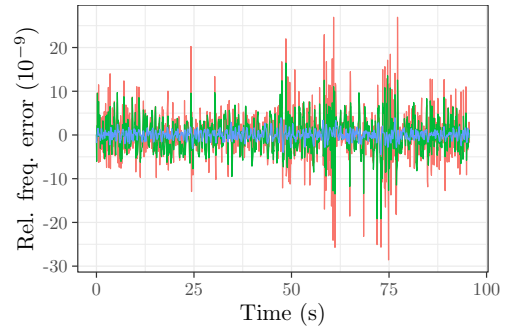


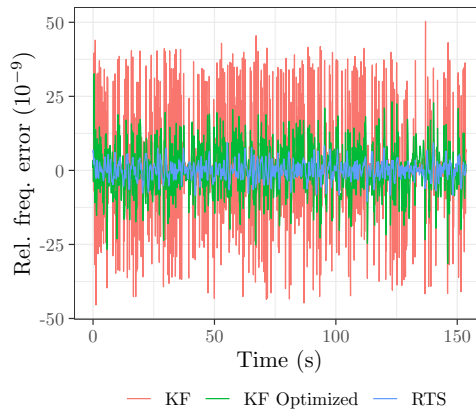
Figure 5.13: Example of the estimation results in a harsh indoor environment. The figure shows the estimated relative frequency minus one (in order to offset the result to zero for better presentation). The red curve shows the ground-truth frequency, while *KF Optimized* is the Kalman filter with the ground-truth noise estimate. Note that the Kalman filter with the optimized noise estimation provides a smaller standard deviation. The figure also shows that each algorithm is able to follow the bias of the frequency in time.



(a) Calm environment



(b) Normal environment



(c) Harsh indoor environment

Figure 5.14: The error between the estimated relative frequency and the ground truth relative frequency of the slave device against time. The Kalman filter with noise estimation results in a smaller standard deviation.

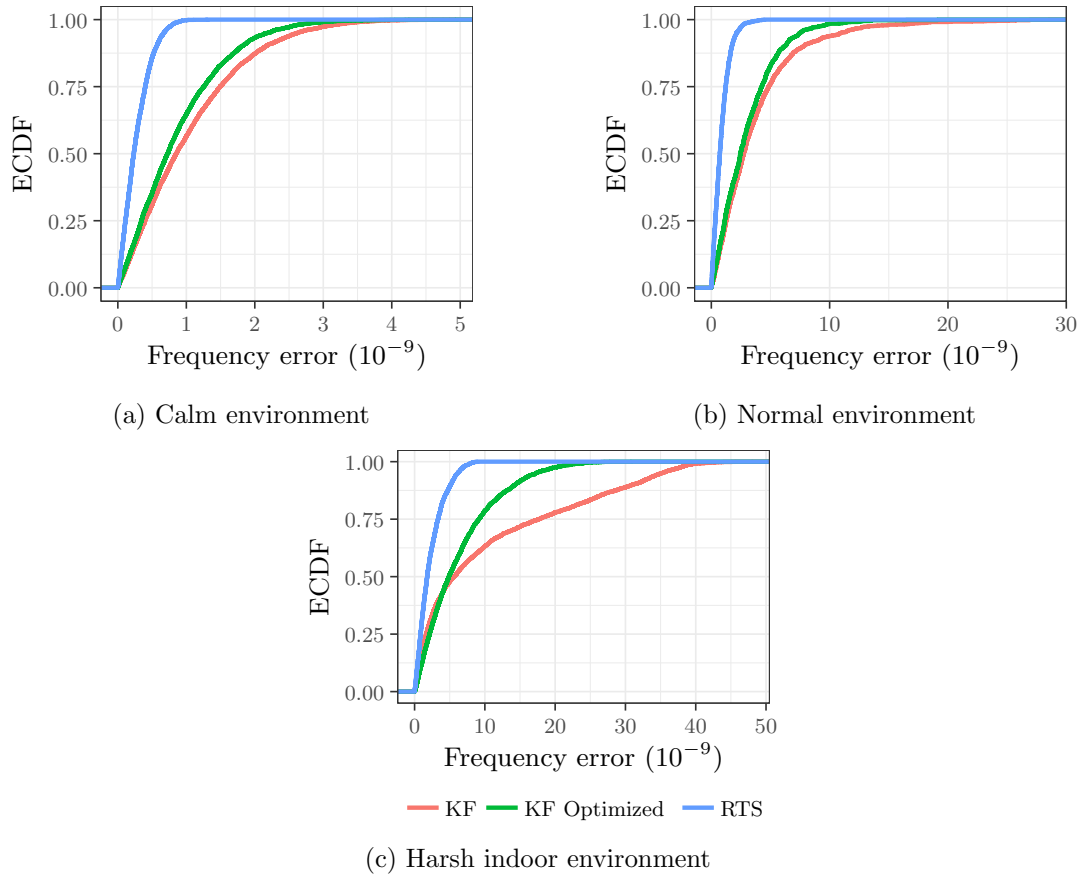


Figure 5.15: The empirical cumulative distribution of the absolute error between the estimated relative frequency and the ground truth relative frequency of the slave device in different measurement scenarios. In all scenarios, the green curve of the noise-optimized Kalman filter gives better results, especially in the case of strong multipath propagation. For reference, the optimal Bayes estimation, the result of the RTS smoother is presented in blue.

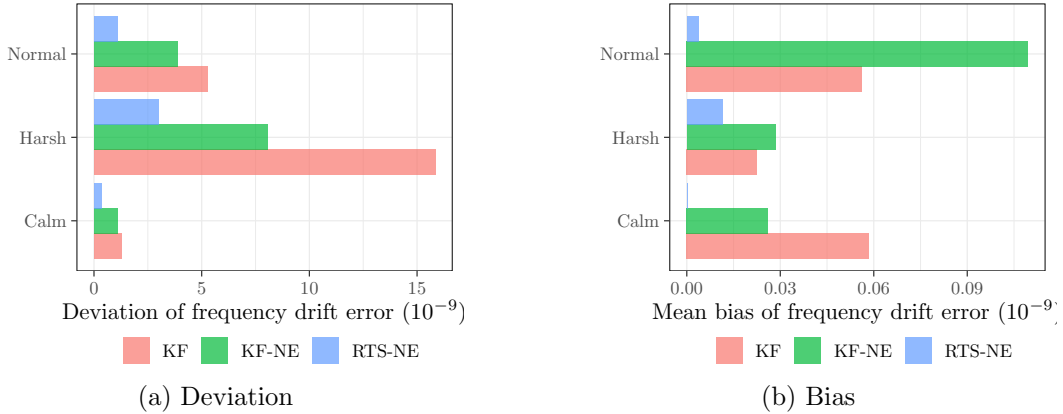


Figure 5.16: The standard deviation and mean bias of the error between the estimated relative frequency and the ground truth relative frequency of the slave device in different measurement scenarios. The colors show different algorithms. Note that in all scenarios, the Kalman filter with the optimized noise outperforms the fixed noise Kalman filter. (NE stands for „Noise Estimation”).

The aggregated results can be seen in Figure 5.16. Here, the standard deviation and mean bias of the frequency drift error are presented for each scenario and method. Note that in each scenario, the improvement in the standard deviation is significant, especially in the case of harsh indoor environments, where it is around half of the original Kalman filter. The bias has a negligible increase in the case of normal and harsh environments. However, the bias is typically three orders smaller than the standard deviation, and the bias following capability is proven for all the algorithms.

5.3.5 Conclusion

The chapter proposed a Bayesian method to estimate the measurement noise variance in a linear dynamic model of clock frequency drift estimation between devices. While the method is evaluated on ultra wide band communication, the results are more general and can be applied in other clock synchronization applications. It has been shown that the the proposed algorithm provides better frequency drift estimations using the optimal noise covariance inferred from input data. In all investigated scenarios, the proposed method resulted in smaller state variance, especially in the case of strong multipath propagation. While the algorithm has a couple of advantages, e.g. the estimation does not require the channel impulse sample from the receiver, the main drawback of the proposed algorithm is the batch nature of the estimation: for inferring the optimal measurement noise, all the measurement needs to be accounted to accomplish the smoothing process. However, the the chain-based architecture of the probability space envisions the possibility to implement the estimation as a filter, estimating the noise variance of the last N measurements, i.e. implementing the method efficiently for a moving window. These methods are closely related to the so-called moving horizon estimator, which pro-

poses a course for further development. Investigating the proposed method in various propagation scenarios (e.g., wired environments or WiFi) is an interesting topic in its own right.

5.4 Thesis summary

Thesis Group 3 – By modeling, simulating, and analyzing real measurements of the distribution of receive timestamps in Ultra Wide Band systems, I have shown, that the distributions are diverse and potentially multimodal. I have proposed an adaptive filter to establish the frequency synchronization between the communicating parties by simultaneously estimating the frequency drift and the measurement noise.

Thesis 3.1 – Using the recommended wide-band channel model of IEEE 802.15.4a standard based on the wide-band Saleh-Valenzuela model, I have shown that the distribution of the error of receive timestamps does not follow a normal distribution by presenting asymmetric parametric distributions with higher likelihood values. [C2]

Thesis 3.2 – I have proposed a method to measure the distribution of the error of receive timestamps up to a constant in real ultra wide band systems, eliminating the effect of frequency drift between transceivers. I have shown that the distributions of the received timestamps are diverse and in some cases, they are not only skewed but multimodal. [C3]

Thesis 3.3 – I proposed a batch method to estimate the variance of the received timestamp error noise from real-life measurements to infer the frequency drift between transceivers. The method applies a third-order state model Kalman-filter along with a Bayesian smoother to estimate the optimal states, from which an Expectation-Maximalization technique estimates the measurement noise. The method increases both the accuracy and the precision of the frequency estimation compared to the classical Kalman filter-based solutions in different measurement scenarios. [C3]

6

Precise time synchronization in IEEE 802.1AS systems

While the IEEE 1588 standard that describes Precision Time Protocol precisely outlines the procedure for calculating the actual clock offset from measurements, it does not specify how implementations actually determine the exact clock parameters. Beyond various protocols and technologies, the core of each implementation lies in the clock synchronization algorithm, which estimates the actual frequency and offset values based on noisy measurements. Although many promising methods employ probabilistic approaches – such as Kalman filters – to estimate the clock offset and frequency, they are suboptimal in that the precise measurement noise or packet delay variation must be known in advance. Insufficiently known a priori filter statistics will, on the one hand, reduce the precision of the estimated filter states, introduce biases to their estimates, or even state divergence [122].

The application of adaptive stochastic filtering, specifically adaptive Kalman filtering, lags behind within the scope of IEEE 1588 despite its demonstrated superior performance in comparison to alternative estimators. The time synchronization issue presents certain advantages over general state estimation problems. For instance, although accurately estimating the Packet Delay Variation (PDV) is crucial, it is possible to precisely determine the process noise in advance. Furthermore, the measurement model with time-invariant noise in time synchronization opens up opportunities for employing diverse adaptive estimation algorithms.

Adaptive filtering – i.e., estimating the measurement noise – is a rarely studied topic regarding the IEEE 1588 protocol. To our knowledge, few works exist on estimating the *theoretically optimal* measurement noise in the context of clock synchronization using probability filtering. The [123] follows the approach of approximate Bayesian filtering, calculating the posterior state density numerically, resulting in optimal state and noise

estimation, although the method is computationally too intensive to be applied in real-time systems.

One approach to adaptive estimation involves Bayesian model averaging, in which parallel models with different noise estimations are employed, and an adaptive averaging method is utilized to deduce the estimated value [124]. Variational Bayes method and [Interacting Multiple Models \(IMM\)](#) can be used to implement adaptive noise estimation, even for non-Gaussian noise [125]. However, there are drawbacks to the use of multiple models. One issue is the suggestion that one of the models should be near the optimal model. Moreover, it has been demonstrated that employing excessively large model banks can diminish performance, as the true model faces too much competition from false models [126].

Another approach involves finding the maximum likelihood solution for the optimal measurement noise covariance. For instance, the Field Kalman Filter (FKF) [127] employs a computationally intensive [BFGS](#) optimization. Alternatively, in [128], an M-robust estimate is used in an iterative non-linear optimization, applying the Huber-loss to address outlier depression and estimate the measurement noise. The [Adaptive Kalman Filter \(AKF\)](#) [129] operates based on the whiteness of the innovation sequence, adjusting the measurement noise covariance matrix to achieve white noise characteristics in the innovation sequence. An iterative method with moderate computational complexity for optimal Bayes estimation can be found in [130], relying on Bayesian learning of parameters in linear dynamic systems through expectation-maximization, which method guarantees the convergence to the optimal estimation.

It is important to note that in real systems, the forward and backward delay also can mismatch, and to handle the so-called packet delay asymmetry, a couple of methods were proposed in the literature (e.g., [131]), and the IEEE 1588 standard provides means to communicate asymmetries to estimation algorithms. Freire et al. [55] gives a comprehensive overview of the methods for estimating the packet delay asymmetry.

The primary contribution of this chapter is the introduction of a measurement model featuring time-invariant noise covariance. Building upon this model, the chapter proposes a computationally feasible Adaptive Kalman Filtering algorithm grounded in optimal Bayesian estimation. This algorithm aims at the online prediction of packet delay variation in the IEEE 1588 protocol. The adaptive approach leads to both enhanced precision, thereby reducing estimation variance, and improved accuracy by mitigating bias. To evaluate the proposed solution, simulations are conducted in a controlled environment using clocks with known Allan variance. Additionally, real-world assessments are performed in a [PTP](#)-unaware environment with diverse background traffic scenarios.

6.1 Adaptive estimation of clock state in PTP clock servo

This section introduces the Adaptive Kalman Filter for real-time estimation of measurement noise in PTP time synchronization applications. Subsequently, the clock dynamic model and the proposed measurement model are outlined, both designed to be compatible with the AKF.

While the clock state estimation along with the PDV noise variance estimation can be handled by the method presented in Chapter 5 based on latent state space models parameter estimation using expectation-maximization, it is computationally intensive thanks to the smoothing process required for optimal estimation. This section proposes an alternative, the Adaptive Kalman filter-based method, which, in fact, has performance very close to optimal solutions, as it will be presented in later sections.

6.1.1 Adaptive Kalman filter

Adaptive Kalman filter is an innovation-based adaptive method that has a long history, and it was successfully applied in various applications [129]. While Kalman-filtering was introduced in Section 2.1, for better understanding, the Kalman-filter equations are presented here from a statistical viewpoint (i.e., concentrating on expectation and variance) rather than using the Bayesian viewpoint. Kalman-filter models a dynamic problem as

$$\begin{aligned}\mathbf{x}_k &= A_k \mathbf{x}_{k-1} + \mathbf{q}_k \\ \mathbf{y}_k &= H_k \mathbf{x}_k + \mathbf{r}_k\end{aligned}\tag{6.1}$$

where A_k is the state-transition matrix, H_k is the observation matrix, \mathbf{x}_k is the state at step k , \mathbf{y}_k is the measurement vector, \mathbf{q}_k and \mathbf{r}_k are uncorrelated, white Gaussian noises, i.e.

$$\begin{aligned}\mathbb{E}[\mathbf{q}_k] &= \mathbf{0} & \mathbb{E}[\mathbf{r}_k] &= \mathbf{0} \\ \mathbb{E}[\mathbf{q}_i \mathbf{q}_j^T] &= Q_i \delta_{ij} & \mathbb{E}[\mathbf{r}_i \mathbf{r}_j^T] &= R_i \delta_{ij}\end{aligned}$$

with R_i the measurement noise covariance matrix and Q_i the process or dynamic model noise covariance matrix, which can depend on the actual time step, and δ_{ij} is the Kronecker-delta function.

Having known A_k, H_k, R_i, Q_i matrices, Kalman filter estimates the mean $\hat{\mathbf{x}}_k = \mathbb{E}[\mathbf{x}_k]$ and the covariance $P_k = \text{Cov}[\mathbf{x}_k]$ of the state variable \mathbf{x}_k in the k th time step (the hat symbol denotes the mean estimate). Having an a priori known $\hat{\mathbf{x}}_0, P_0$, the optimal state estimations in least squares sense for $k > 0$ are the Kalman filtering equations:

$$\begin{aligned}\hat{\mathbf{x}}_k^- &= A_{k-1} \hat{\mathbf{x}}_{k-1} \\ P_k^- &= A_{k-1} P_{k-1} A_{k-1}^T + Q_k\end{aligned}\tag{6.2}$$

and the measurement step

$$\begin{aligned}\mathbf{v}_k &= \mathbf{y}_k - H_k \hat{\mathbf{x}}_k^- \\ S_k &= H_k P_k^- H_k^T + R_k \\ K_k &= P_k^- H_k^T S_k^{-1} \\ \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + K_k \mathbf{v}_k \\ P_k &= (I - K_k H_k) P_k^-\end{aligned}\tag{6.3}$$

Note, that the matrix S_k is the estimate of the covariance of the innovations \mathbf{v}_k , which have a central role in the AKF. Also, Kalman filtering assumes that $\mathbb{E}[\mathbf{v}_k] = 0$.

Suppose now, that R_k does not depend on the timestep, i.e. $R_k \equiv R$. In the steady state of the filter, $S_k \rightarrow S$, the probability density of the measurements can be described as a multivariate normal distribution

$$p(y_k|\alpha) = \frac{1}{\sqrt{(2\pi)^m |S|}} e^{-\frac{1}{2} \mathbf{v}_k^T S^{-1} \mathbf{v}_k}$$

where m is the dimension of the measurement vector, and α is the set of any parameter in the estimation problem, e.g., the R covariance matrix. For a window of $i = 1, \dots, W$ consecutive measurements, this results in the maximum likelihood problem

$$\hat{\alpha} = \arg \min_{\alpha} \sum_{i=1}^W \ln |S| + \sum_{i=1}^W \mathbf{v}_i^T S^{-1} \mathbf{v}_i \quad (6.4)$$

Using matrix calculus, the derivative of (6.4) is (see [129])

$$\sum_{i=1}^W \text{tr} \left\{ (S^{-1} - S^{-1} \mathbf{v}_i \mathbf{v}_i^T S^{-1}) \frac{\partial R(\alpha)}{\partial \alpha} \right\} = 0 \quad (6.5)$$

where we marked the dependence of R on the α parameters. Assuming, that R is diagonal, and $\alpha_i = R_{ii}$ (e.g. the diagonal entries), yields

$$\sum_{i=1}^W \text{tr} \{ S^{-1} (S - \mathbf{v}_i \mathbf{v}_i^T) S^{-1} \} = 0 \quad (6.6)$$

which implies that the S innovation covariance matrix can be estimated as

$$\hat{S} = \frac{1}{W} \sum_{i=1}^W \mathbf{v}_i \mathbf{v}_i^T \quad (6.7)$$

Please note that the Kalman filter also has an estimate on S , namely $S_k = H_k P_k^- H_k^T + R_k$. Combining the expression of S_k in (6.3) and (6.7), the estimated \hat{R} measurement covariance matrix is

$$\hat{R} = \hat{S} - H_k P_k^- H_k^T \quad (6.8)$$

since the two expression shall be equivalent. Note that this method estimates the measurement covariance matrix by *covariance matching*, i.e. the measurement covariance is corrected by the difference of the Kalman filter estimate of the innovation covariance and the covariance calculated from the innovations.

Please note that the computational complexity of the algorithm is really low – it only requires estimating the covariance of the innovations in a W wide window, which can be implemented easily in real-time environments. However, two notes shall be made on the applicability of the method:

1. The method does not enforce any positive definite constraint on the estimated covariance matrix, sometimes resulting in a matrix that is itself not a covariance matrix. In this case, the estimate can be ignored.
2. The AKF is strongly dependent on the initial estimate of the R matrix. Multiple order differences in the initial covariance estimate and the real measurement noise covariance result in stability issues, which requires the implementation to provide at least a coarse estimate of the initial noise covariance.

6.1.2 Time synchronization model

This subsection presents the main aspect of the time synchronization model applied and also covers the time-invariant measurement noise model. Basically, for the timestamp exchange procedure, the standard PTP protocol is assumed (see (2.20)) and the clock is modeled as (2.14), with only two states (i.e. $x(t)$ and $\dot{x}(t)$) and with $\sigma_3 = 0$.

The measurement models in [62, 132] assume that both the clock offset and the clock skew can be observed, resulting in a time-variant measurement noise with a covariance matrix not being diagonal. The fact that the AKF requires the measurement covariance matrix to be diagonal and time-invariant, a new measurement model is proposed. As previously stated, the clock model here models the clock phase directly, so the measurement model proposed uses only the clock phase observation, which also results in observable instantaneous frequency.

In theory, the measurement model is simple (see Figure 2.6), since the actual time in the k th synchronization period at the slave when the *Sync* message arrives is

$$x_k = t_k^{(1)} + t_k^{PD} \quad (6.9)$$

However, the calculation of the packet delay, i.e., t_k^{PD} , is not straightforward. The first issue is that the packet delay is a random variable, i.e. it can differ from period to period. The reason behind the random behavior of packet delay is that the receive timestamps (specifically $t^{(2)}$ and $t^{(4)}$) have delay uncertainty, e.g. from queuing delays. To consider this uncertainty, receive timestamps are modeled as the sum of a ground-truth value and noise. Let the ground-truth value of $t^{(2)}$ be $\overline{t^{(2)}}$, and the measurement is $t^{(2)} = \overline{t^{(2)}} + \xi_{PDV}^{(2)}$. Also, $t^{(4)} = \overline{t^{(4)}} + \xi_{PDV}^{(4)}$, where $\xi_{PDV}^{(2)}, \xi_{PDV}^{(4)}$ are random variables.

From Figure 2.6, the packet delay can be written as

$$\begin{aligned} t_k^{PD} &= \frac{(\overline{t^{(4)}} - t^{(1)}) - (t^{(3)} - \overline{t^{(2)}})}{2} \\ &= \frac{(t^{(4)} - t^{(1)}) - (t^{(3)} - t^{(2)})}{2} - \frac{\xi_{PDV}^{(4)} + \xi_{PDV}^{(2)}}{2} \end{aligned} \quad (6.10)$$

Notice that the uncertainties in the receive timestamps do not cancel each other, so it can lead to systematic bias in the estimation of the packet delay. This phenomenon is referred to as *delay asymmetry* between the master-to-slave and slave-to-master delay in the literature.

The second issue is that the $(t^{(3)} - t^{(2)})$ expression is also a random variable in the estimation of the packet delay. Neither $t^{(2)}$ nor $t^{(3)}$ can be measured; instead, $\tau^{(2)}, \tau^{(3)}$ are recorded by the local clock. Using the same method, as in Section 2.2.3 (i.e. integrating from $\tau_k^{(2)}$ to $\tau_k^{(3)}$), the distribution of $(t_k^{(3)} - t_k^{(2)})$ can be written as

$$(t_k^{(3)} - t_k^{(2)}) \sim \mathcal{N} \left(\dot{x}_k(\tau_k^{(3)} - \tau_k^{(2)}); \sigma_1^2(\tau_k^{(3)} - \tau_k^{(2)}) + \sigma_2^2 \frac{(\tau_k^{(3)} - \tau_k^{(2)})^3}{3} \right) \quad (6.11)$$

Combining (6.9), (6.10) and (6.11), measurement model can be written as

$$\frac{t_k^{(1)} + t_k^{(4)}}{2} = H_k \mathbf{x}_k + r_k \quad (6.12)$$

with

$$H_k = \left[1 \quad \frac{1}{2}(\tau_k^{(3)} - \tau_k^{(2)}) \right] \quad r_k = \frac{\varepsilon}{2} + \frac{\xi_{PDV}^{(4)} + \xi_{PDV}^{(2)}}{2}$$

and $\varepsilon \sim \mathcal{N}(0; \sigma_\varepsilon^2)$ with $\sigma_\varepsilon^2 = \sigma_1^2(\tau_k^{(3)} - \tau_k^{(2)}) + \sigma_2^2 \frac{(\tau_k^{(3)} - \tau_k^{(2)})^3}{3}$. The ε noise is the effect of the uncertainty of the drift of the local clock while estimating the real delay between $t^{(2)}$ and $t^{(3)}$, while noises $\xi_{PDV}^{(4)}, \xi_{PDV}^{(2)}$ are the result of the packet delay variation. Note that the measurement noise depends on the time step only through the response time, and the three noise sources are considered to be independent, so the variance can be calculated as

$$\begin{aligned} R = \text{Var} [r_k^2] &= \text{Var} \left[\frac{\varepsilon}{2} \right] + \text{Var} \left[\frac{\xi_{PDV}^{(4)} + \xi_{PDV}^{(2)}}{2} \right] = \\ &= \sigma_\theta^2 \frac{\tau_k^{(3)} - \tau_k^{(2)}}{4} + \sigma_\gamma^2 \frac{(\tau_k^{(3)} - \tau_k^{(2)})^3}{12} + \frac{\sigma_\xi^2}{2} \approx \frac{\sigma_\xi^2}{2} \end{aligned} \quad (6.13)$$

Since the response time in real systems is in the order of milliseconds, the contribution of the clock noise ε can be ignored, resulting in a measurement noise effectively being time-invariant. For the approximation, we assume that $\text{Var}[\xi_{PDV}^{(4)}] = \text{Var}[\xi_{PDV}^{(2)}] = \sigma_\xi^2$, and the noises are independent.

While classic Kalman filter-based solutions (e.g. [62, 132]) use both the offset and the estimated skew as measurements, however, the proposed model utilizes only the received timestamp. To present, that both states are observable in (6.12), we need to show, that the observability matrix

$$O = \begin{bmatrix} H_k \\ A_k H_k \end{bmatrix} = \begin{bmatrix} 1 & \frac{1}{2} \Delta_k \\ 1 & \frac{3}{2} \Delta_k \end{bmatrix}$$

has full-rank, which is readily visible for every $\Delta_k \neq 0$. Note, that as earlier in the dynamic model (2.15), $\Delta_k = t_k - t_{k-1}$.

6.2 Evaluation of proposed method

The performance of the the algorithm was proven both by simulation and real-world measurements. For comparison, different algorithms were also evaluated on the very same dataset. For a baseline method, the offset calculation of the IEEE 1588-2019 [11] standard was applied. For comparison, different state-of-the-art methods were evaluated, as follows:

TH The theoretically optimal Kalman filter, using the models in (2.15),(6.12) and the theoretical measurement noise variance in (6.13)

KF The Kalman-filter defined in [62] with measurement noise, $\sigma_{\theta_M} = 10^{-7}$ s (where σ_{θ_M} is the measurement noise), using the original process and measurement model

KF-HN Also [62] with a much higher $\sigma_{\theta_M} = 10^{-4}$ s noise standard deviation

M-robust Robust and adaptive Kalman filter defined in [128], using the models in (2.15),(6.12)

KF-EM Expectation-maximization based optimal Kalman filter defined in Chapter 5, using the models in (2.15),(6.12)

MMax Minimax technique [56] with median statistics

PTP-LP PTP-LP linear programming based method using the PTP-H heuristics [58]

Please note that the measurement noise standard deviations were selected arbitrarily in the case of the classical Kalman filtering methods (i.e., KF and KF-HN). For evaluation, a high measurement noise standard deviation (0.1 ms) and a low measurement noise standard deviation (100ns) were selected, resulting in a high and a low noise suppression behavior, respectively.

The simulations were implemented in R language, using the *dynamic clock model* defined by (2.15), with typical values of $\sigma_1 = 10^{-6}$ s and $\sigma_2 = 10^{-8}$. In the simulation, the measurement noises $\xi_{\text{PDV}}^{(2)}$ and $\xi_{\text{PDV}}^{(4)}$ were independent with identical standard deviation ranging from $\sigma_\xi = 10^{-2}$ s to $\sigma_\xi = 10^{-9}$ s, where σ_ξ is the standard deviation of the noise process. Since arrival processes can be modeled with exponential distribution, the algorithms were evaluated using both Gaussian and exponential distributions¹ (resulting in a measurement noise with gamma distribution) using the dynamic clock model. In the simulation, the response delay (i.e. $\tau^3 - \tau^2$) mean was 10ms. The simulation time step was $\Delta = 1$ s, and was running to $t_{\text{end}} = 10000$ s.

In the evaluation of the adaptive algorithms (AKF, KF-EM, M-robust), a $W = 20$ wide window was applied. The initial measurement noise variance was $R_0 = 9 \cdot 10^{-14} s^2$. To stabilize the AKF algorithm, the initial measurement noise estimate was estimated

¹In the case of the exponential distribution, variance is considered to be $\sigma^2 = \lambda^{-2}$.

by the variance of the IEEE 1588-2019 offset measurement variance, namely

$$\hat{R}_0 = \frac{1}{W} \sum_{i=1}^W \Delta t_i^2$$

where Δt is the time offset in (2.20).

To validate the algorithm on real measurements, the IEEE 1588 protocol was chosen to provide measurements for the clock synchronization. To provide a variety of measurement noises (both in distribution and variance), we tested the solution over a PTP-unaware network, using different measurement scenarios and measurement setups. The measurements were done end-to-end (E2E) using a PTP grand master and a PTP slave device. Since the relative running of the clocks of two or more devices is unknown, it is hard to acquire the ground-truth values for the results to be comparable. To overcome this issue, we used two ports of an ESPRESSObin v7 device to be the PTP master and the PTP slave since ESPRESSObin has one and the same clock for all PTP ports. The only problem was to trick the Linux operating system into sending the packets onto the wire instead of a local shortcut, for which simple firewall and ARP rules were applied (see Listing 6.1). This way, the ground-truth time at the reception of a *Sync* message is the same as the RX timestamp, providing ground-truth values for the clock estimation. The hardware clock of the ESPRESSObin runs on 125MHz providing 8ns clock resolution. The measurement setup provides no bias between the devices since they use the very same clock. This way, even a low-pass filter with cut-off frequency $f_c \rightarrow 0$ would provide excellent results. To present the bias following capability of the solutions, artificial bias was added using the dynamic model (2.15), selecting $\sigma_1 = 10^{-6}$ s and $\sigma_2 = 10^{-8}$.

```

ifconfig eth0 10.50.0.1/24
ifconfig eth1 10.50.1.1/24

iptables -t nat -A POSTROUTING -s 10.50.0.1 -d 10.60.1.1 -j SNAT
                                     --to-source 10.60.0.1
iptables -t nat -A PREROUTING -d 10.60.0.1 -j DNAT --to-destination 10.50.0.1
iptables -t nat -A POSTROUTING -s 10.50.1.1 -d 10.60.0.1 -j SNAT
                                     --to-source 10.60.1.1
iptables -t nat -A PREROUTING -d 10.60.1.1 -j DNAT --to-destination 10.50.1.1

ip route add 10.60.1.1 dev eth0
arp -i eth0 -s 10.60.1.1 $IFACE2.MAC

ip route add 10.60.0.1 dev eth1
arp -i eth1 -s 10.60.0.1 $IFACE1.MAC

```

Listing 6.1: Configuration of ESPRESSObin v7 board to enforce packets to be forwarded to the wire and skipping local loopback.

The validation was carried out in 6 different scenarios. Two switches were applied, namely a Cisco SF100D-16P and a Netgear GS108T. There were scenarios utilizing one switch and scenarios using both switches. In scenarios with two switches, the master is connected to the Cisco device while the slave is connected to the Netgear switch, and

the background traffic is enforced between the switches. The 6 scenarios differ in the network setup and in the amount of background traffic. The scenarios are as follows:

1. Using the Cisco switch to connect the master and the slave, no background traffic.
2. Using the Cisco switch to connect the master and the slave with 100 Mbps independent bidirectional background traffic.
3. Using two switches with independent 100 Mbps bidirectional background traffic.
4. Using two switches with independent 50 Mbps bidirectional background traffic.
5. Using two switches without background traffic.
6. Using the department LAN network between ESPRESSObin's ports on a typical workday.

Figure 6.1 shows the distributions of the packet delays separately for the *Sync* and the *Delay response* messages. It is clearly visible that the scenarios provide rather different packet delay distributions, both in variance and shape. E.g., Scenario 4 shows a strong asymmetric distribution, as can be seen on its boxplot, while in Scenarios 1 and 2, *Sync* messages have nearly uniform delay distributions with different variances. Also, it is clear that the packet delays are strongly asymmetric in the sense of the IEEE 1588, i.e., *Sync* and *Delay response* messages have different expected delays. However, the focus is on the variance mainly, so to handle the issue of asymmetry, the numerous methods proposed in the literature can be applied.

Using Gaussian packet delay variation in simulation, the variance and the bias of the time offset estimation can be seen in Figure 6.2. The KF-EM provides a variance close to the theoretical one. However, the AKF solution provides similar results except for a very small noise variance ($\sigma_1 < 10^{-6}$ s). In the evaluation, at very small PDV noises, the AKF tends to estimate negative variance, which is ignored, and the previous value has been applied. Also, the time invariance of the approximation in (6.13) becomes invalid when σ_ξ is in the order of tens of nanoseconds. It is worth noting that small variances can be compared to the variance of the clock noise itself since the Allan deviation of the clock is $\sigma_y(\tau = 1\text{s}) \approx 10^{-6}$ s. Also, the AKF provides very good results in estimating the bias, providing close to the theoretical value, and at high noises, it proves more precise than the theoretically optimal KF-EM. Non-adaptive Kalman filters provide sufficient performance only at the measurement noise fitted to the noise parameter, especially the KF-HN, which fails to follow the change in bias at low noise values. All other solutions are behind in performance compared to the AKF and KF-EM. However, KF-EM has higher computational complexity compared to the AKF since it requires multiple forward-backward filtering at each time step. The simulation results for exponentially distributed measurement noise can be seen in Figure 6.3, showing similar results as in the case of normally distributed noise.

The Allan-variance plots for $\sigma_\xi = 1 \times 10^{-3}$ s and $\sigma_\xi = 1 \times 10^{-7}$ s can be found in Figure 6.4 and 6.5, respectively. In both situations, the AKF provides performance

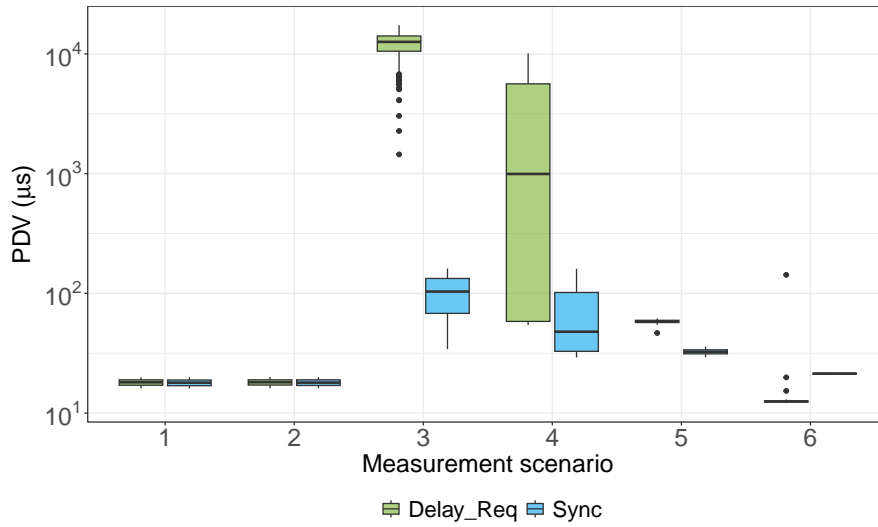


Figure 6.1: Packet delay distributions (PDV) in different scenarios as box plots (note the logarithmic axis). In most cases, the forward-backward distributions are strongly asymmetric with extensive differences in noise variance.

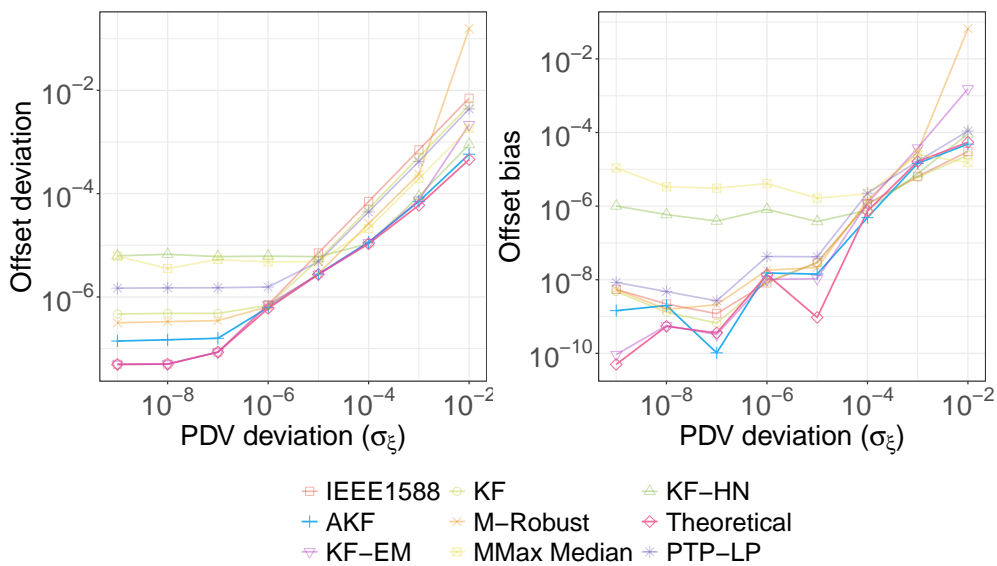


Figure 6.2: The standard deviation (left) and bias (right) of the master-slave offset (both in seconds) estimation at different measurement noise standard deviations using a normal distribution ($\sigma_1 = 10^{-6}$, $\sigma_2 = 10^{-8}$).

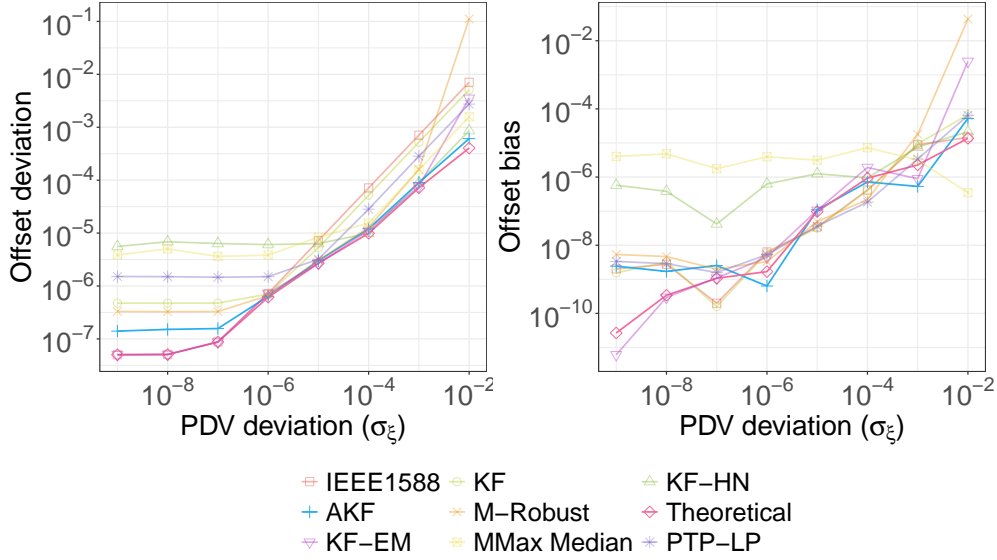


Figure 6.3: The standard deviation (left) and bias (right) of the master-slave offset (both in seconds) estimation at different measurement noise standard deviations using exponential distribution ($\sigma_1 = 10^{-6}$, $\sigma_2 = 10^{-8}$).

close to the theoretical value; however, for high measurement noises, the KF-EM, M-Robust, and KF-HN also provide good performance. For small averaging time spans, KF-HN provides excellent performance (since the Allan variance is close to or better than the theoretical one), but this results in high bias, as has been previously shown. Also, KF-HN provides very poor performance when the measurement noise is small – in these situations, AKF provides good performance, and the KF-EM provides close to theoretical performance but with much higher computational complexity.

The noise estimation accuracy of the adaptive methods can be found in Figure 6.6. At small measurement noises, the AKF overestimates the theoretically optimal measurement noise, mainly because of the assumption of the time invariance of the measurement covariance invalidates in (6.13). The M-robust method systematically underestimates the measurement noise. Also, it is worth noting that the KF-EM overestimates the measurement noise at very high PDV noises and underestimates with multiple orders at low PDV noises. This difference between KF-EM and AKF results from the fact that the KF-EM provides the theoretically optimal estimation, and at low PDV noises, the dominance of clock process noise drives the KF-EM to place complete trust in the measurements.

For real measurements, the Allan variance plots for Scenario 1 and 3 are plotted in Figure 6.7. For Scenario 3, the M-Robust method has some stability issues, resulting in high Allan variance at long averaging time spans. For Scenario 3 (which contains high PDV noise), the KF-HN provides close to theoretical performance; however, it fails to provide it at low PDV noises. The AKF and the KF-EM provide similar performance

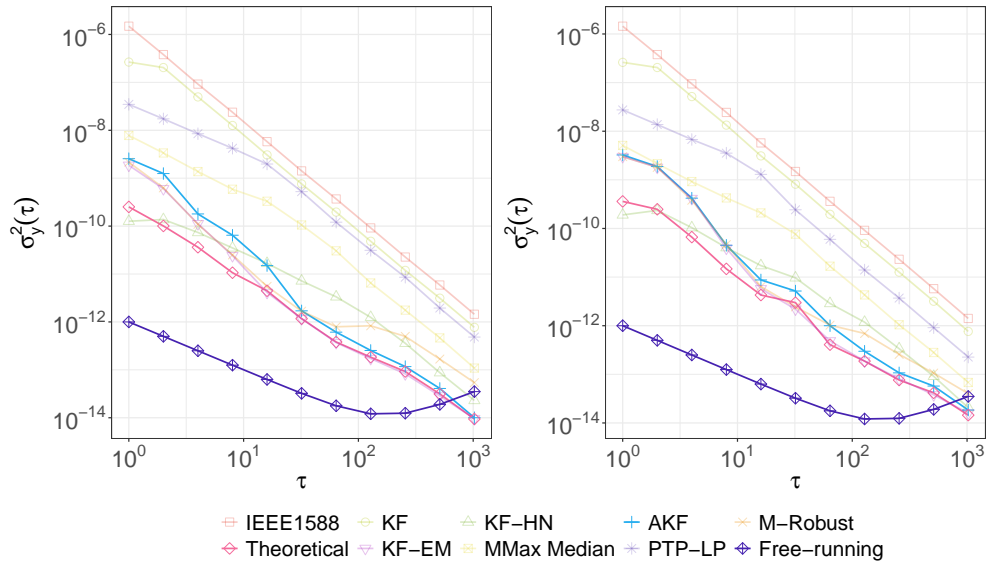


Figure 6.4: The overlapped Allan variance (in s^2) plots of the estimators in simulation for normally (left) and exponential (right) distributed measurement noise with $\sigma_\xi = 10^{-3}s$. For reference, the plots depict the theoretical Allan deviation of the free-running clock. ($\sigma_1 = 10^{-6}$, $\sigma_2 = 10^{-8}$)

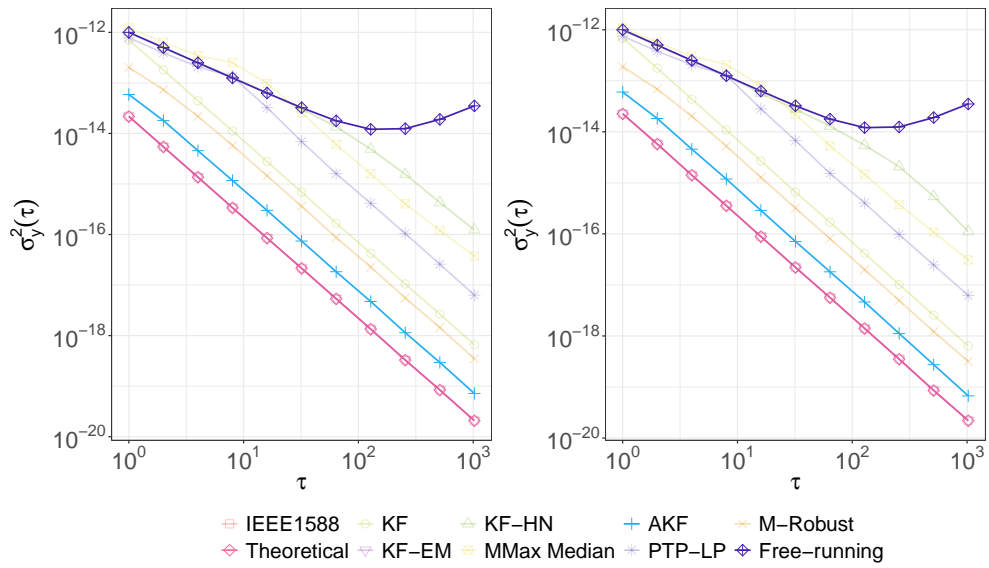


Figure 6.5: The overlapped Allan variance (in s^2) plots of the estimators in simulation for normally (left) and exponential (right) distributed measurement noise with $\sigma_\xi = 10^{-7}s$. For reference, the plots depict the theoretical Allan deviation of the free-running clock. ($\sigma_1 = 10^{-6}$, $\sigma_2 = 10^{-8}$)

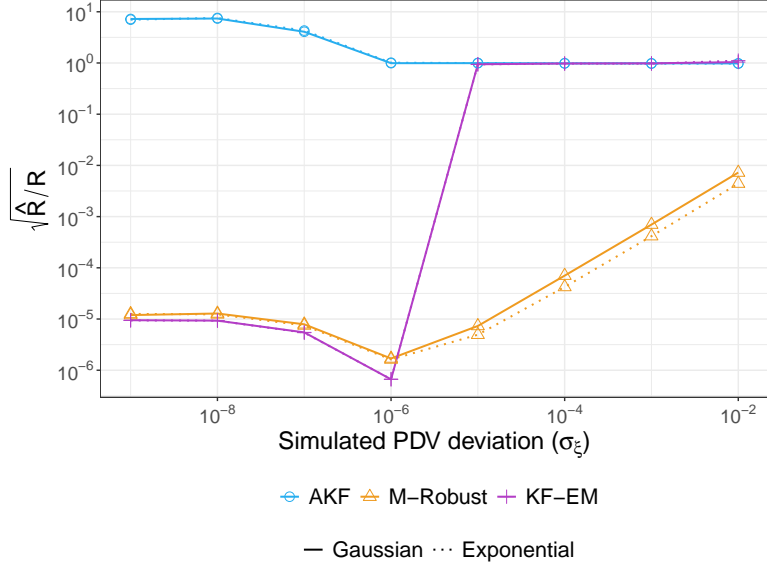


Figure 6.6: The estimated value and the ground-truth value of the measurement noise standard deviation as a ratio (i.e. $\sqrt{\hat{R}/R}$, see (6.13)) for the adaptive methods at different ground-truth PDV noises.

in both scenarios. Spider charts in Figure 6.8 and 6.9 also show the superiority of the AKF and KF-EM algorithms, mostly in scenarios with high PDV noise, like Scenario 3 and 4.

6.2.1 Remarks on computational complexity

Classical Kalman filters do a couple of matrix multiplications per measurement. However, the computational complexity can depend on the state and measurement dimension. Let the state $\mathbf{x} \in \mathbb{R}^n$ and the measurement $\mathbf{y} \in \mathbb{R}^m$, then the computational complexity is $\mathcal{O}(n^{2.4} + m^2)$ [133]. The dimensions can be considered fixed (i.e., 2D) for all the algorithms presented. So, the running time can be considered constant in case a new measurement arrives for a specified dynamic and measurement model.

For the Adaptive Kalman Filter, an additional measurement covariance estimation calculation is performed (see (6.8)), which depends on the W window size. There, only the covariance calculation contains the W window size (see (6.7)), yielding a computational complexity for W to $\mathcal{O}(W)$, which simply means a linear dependence on the window size.

Theoretically supporting the computational complexity of the algorithms is challenging since each method involves different steps to compute results. The comparison is especially hard since their dependence on different parameters can mislead the comparison. For most algorithms, the complexity can be regarded as constant time since they perform a fixed number of calculations per measurement. However, an exception arises

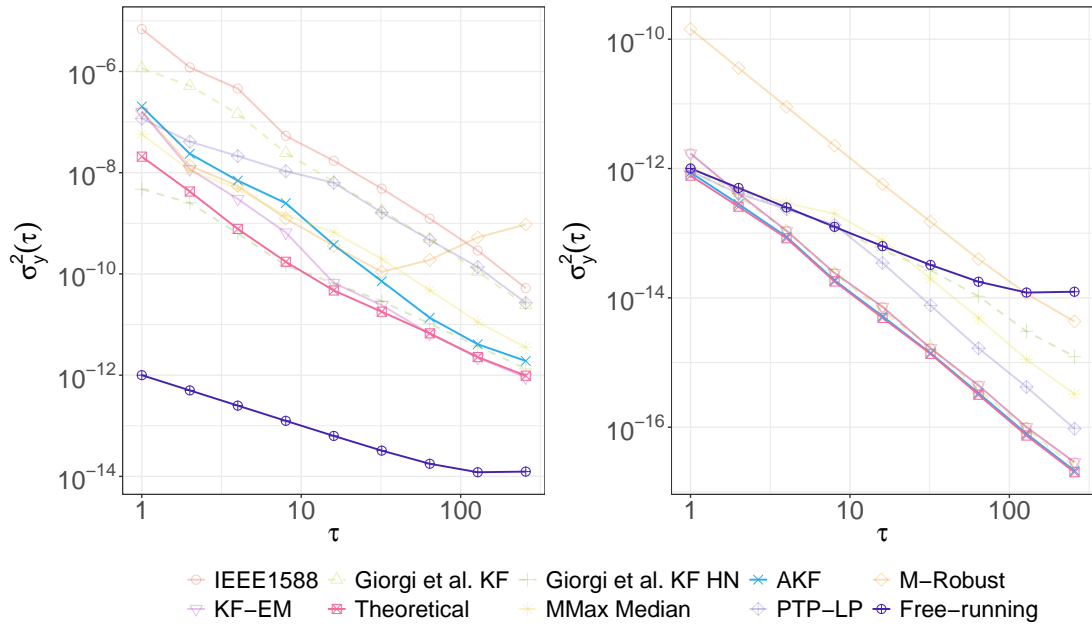


Figure 6.7: The overlapped Allan variance plots (in s^2) of the estimators in real measurement for Scenario 3 (left) and Scenario 1 (right). For reference, the plots depict the theoretical Allan deviation of the free-running clock. ($\sigma_1 = 10^{-6}$, $\sigma_2 = 10^{-8}$)

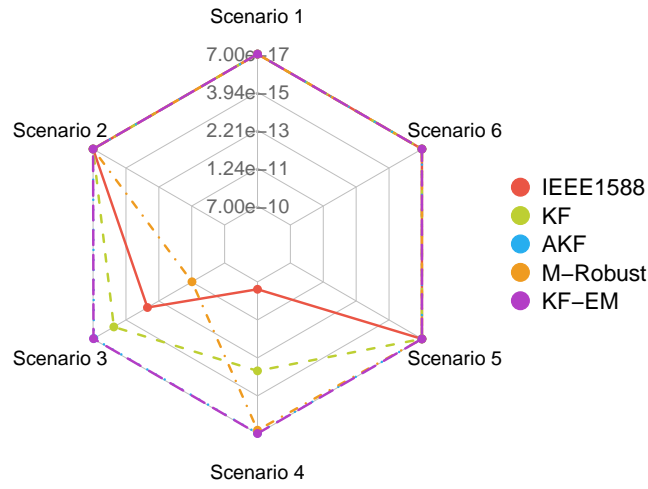


Figure 6.8: Overlapped Allan variance of different estimators at different measurement scenarios at $\tau = 128s$.

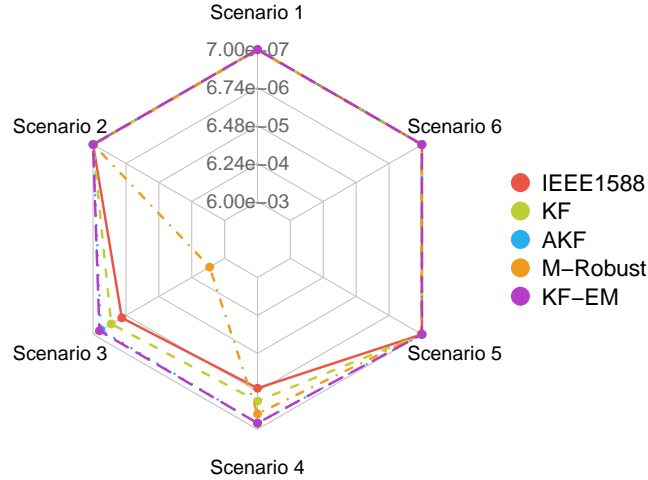


Figure 6.9: Standard deviation of the master-slave offset estimates of different estimators at different measurement scenarios.

when an algorithm uses a window-based approach to adaptively estimate a hyperparameter; in such cases, the complexity depends on the window size W . In this case, the algorithm depends on the W window size. However, while precise analysis of the time complexity is ignored, it is valuable to explore the factors influencing their running times to better understand their complexity. The standard IEEE 1588 algorithm performs only basic operations (e.g., subtraction, division) to estimate the clock state, enabling high-speed computation. Similarly, the Minimax technique involves calculating simple statistics over the last W measurements, making it computationally efficient. In contrast, Kalman filter-based methods (TH, KF, KF-HN) involve a single step of Kalman filtering to compute the current state, with computational complexity depending on the fixed state dimension for each method. However, the M-robust algorithm requires identifying outliers over a window of size W using a minimization process with a nonlinear, robust score function, significantly increasing its computational demand. The high complexity of the KF-EM algorithm comes from the estimation of the noise profile through repeated applications of a smoother and Kalman filter across a W -wide window, making it computationally intensive. On the other hand, the Adaptive Kalman Filter (AKF) introduces only a few additional matrix multiplications, resulting in a computation speed comparable to standard Kalman filtering methods. The PTP-LP algorithm, another window-based approach, performs two linear regressions per measurement to estimate the clock state. Linear regression involves matrix multiplication and inversion, which ensures a relatively high computational speed for PTP-LP.

Please note that implementation details also influence performance. For example, the M-robust algorithm uses the Python-based Robust Kalman Filter library², which is

²<https://github.com/milsto/robust-kalman>

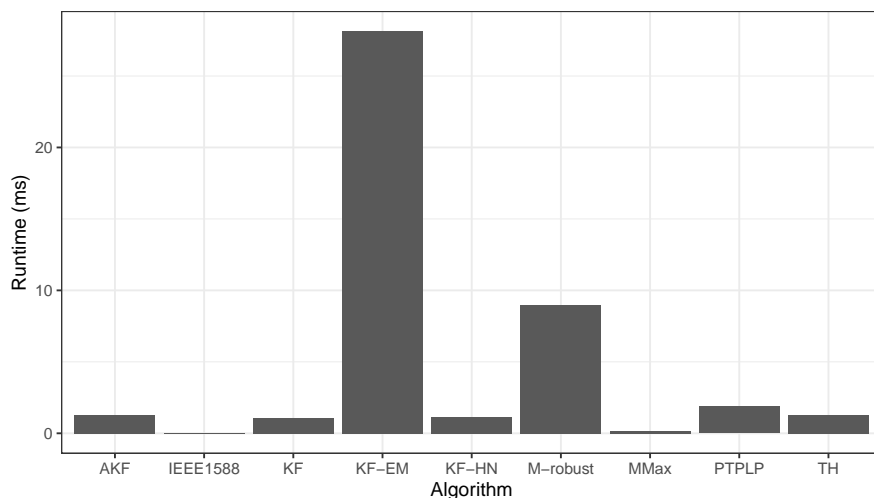


Figure 6.10: Average runtime per estimation step (in milliseconds) for the algorithms, based on a long-run evaluation of 10,000 measurements. Note that the R implementation is not performance-optimized but allows for a relative comparison of running times.

generally faster than R-based implementations. Meanwhile, the PTP-LP algorithm employs the built-in `lm` function in R, which provides efficient linear regression calculations, further contributing to its computational efficiency.

Figure 6.10 presents the runtime performance of the algorithms, measured as the time required to execute a single estimation step (per measurement). It is important to note that the algorithms were implemented in R, a language not optimized for performance, so the reported running times are intended only for rough comparison. Additionally, the algorithms themselves could be further optimized for improved efficiency. The fastest algorithm is the basic offset calculation proposed by the IEEE 1588 standard, with the MMax method offering comparable performance. However, while MMax achieves moderate accuracy relative to other methods, it is almost as fast as the basic IEEE 1588 offset estimation. The KF-EM method, though computationally intensive and slow, delivers near-optimal accuracy. Notably, the AKF method does not demand significantly more runtime compared to standard Kalman filter-based methods (e.g., KF) but provides much better accuracy.

6.3 Thesis summary

Thesis Group 4 – I have derived the time-invariant measurement model of Precision Time Protocol in order to apply adaptive Kalman filtering for synchronizing remote clocks. I have shown, that adaptive Kalman filtering outperforms existing Kalman filtering-based algorithms for Precision Time Protocol both in accuracy and precision.

Thesis 4.1 – I have proposed a method to measure the distribution of the error of received timestamps using the Precision Time Protocol (PTP). My analysis shows that the range of these distributions depends on both the network configuration and its operation. The analysis reveals that even under typical operating scenarios, the variance of the distributions can vary significantly in magnitude, making accurate a priori estimation particularly challenging. [J2]

Thesis 4.2 – I have proposed a probabilistic measurement model for Precision Time Protocol with time-invariant measurement noise variance to provide compatibility with the Adaptive Kalman Filter (AKF). I have shown, that in the proposed model both state variables are observable. [J2]

Thesis 4.3 – I have proposed an online time synchronization algorithm based on Adaptive Kalman Filtering using the measurement model of Thesis 4.2. I have compared the Adaptive Kalman Filtering-based algorithm with different state-of-the-art clock state estimation algorithms, and I have shown that the AKF has a performance close to the optimal filtering with a computational complexity close to classic Kalman filtering. [J2]

7

Applicability of the results and future outlook

Since most of the outcomes from the research efforts are mainly applied research, which emphasizes practical applications by serving specific needs in both industry and technology development. For instance, the findings from Thesis groups 1 and 3 were directly applied to a project titled „Development of mass sport supporting sensor device and analytical service” (2018-1.1.1-MKI-2018-00075). The aim of this project was to enhance the performance of tracking systems by improving measurement frequency and optimizing the overall tracking capabilities. These innovations play a significant role in advancing technologies that support large-scale sports activities, ultimately providing more precise data collection and analysis. Namely, the TDoA (time difference of arrival) localization algorithm implemented requires precise frequency synthonization between anchor tags to accurately estimate locations of the client tags (Thesis group 3). Also, the results of Thesis group 1 help the implemented system to gather more measurements with improved frequency, resulting in more seamless and accurate positioning.

On the other hand, the results from Thesis groups 2 and 4 hold wider implications, particularly in the domain of clock state estimation, which is crucial in clock servo implementations. The importance of accurate clock synchronization cannot be overstated in fields such as telecommunications, networked control systems, and various time-sensitive applications where performance depends on reliable timekeeping. The findings have the potential to significantly contribute to projects such as the PTPd project [134] and The Linux PTP Project [135], both of which are fundamental in the precise synchronization of clocks in distributed systems. However, the primary challenge lies in the architecture of the Linux PTP Project’s `ptp4l` software. Although it offers interfaces for various clock servos, it preprocesses timestamps by calculating offset and skew values. However, the solution proposed in Thesis group 4 relies on raw timestamps. This leaves two options: either adapt the algorithm to work with an alternative, though potentially non-time-

invariant, measurement model or modify the clock servo interface of the `ptp4l` software. In the future, both approaches might be feasible and successful approaches.

From an implementation perspective, the findings of Thesis groups 1 and 3 have already been realized using cost-effective hardware based on the Qorvo DWM1000 module, offering outstanding localization capabilities. While Thesis group 2 explores an interesting but less application-focused topic, it also utilizes affordable hardware and readily available technologies. Furthermore, the work from Thesis group 4 stands out due to its low computational complexity, which is particularly advantageous when considering resource-constrained environments. This characteristic makes it feasible to implement their discoveries in embedded systems that have limited processing power and memory. As a result, these solutions can be deployed in a variety of small, portable devices without compromising on efficiency or performance, expanding their applicability to numerous embedded systems across various industries, including automotive, industrial automation, and consumer electronics.

Future research offers numerous opportunities in the field of robust and optimal clock state estimation. Promising advancements in adaptive filtering, such as the Optimized Kalman Filter [136], provide new ways for further exploration. Beyond measurement noise, investigating the joint estimation of state noise (e.g., clock parameters) presents a valuable research direction. While noise estimation effectively minimizes state variance and enhances accuracy, the presence of outliers poses a significant challenge, steering interest to robust estimation methods [137, 138]. Additionally, asymmetric delay estimation remains a persistent issue in packet-switched networks [139]. From an application standpoint, there is potential to replace UWB technology with the similarly performing 5G Frequency Range 3 radio, which could offer comparable propagation and timestamping capabilities for the use cases discussed in this dissertation.

8

Summary

In computer science, time synchronization possesses a rich history, featuring various techniques and protocols designed to achieve precision down to the nanosecond range. This dissertation has aimed to address specific gaps within the expansive domain of time synchronization.

The prominent protocol for time synchronization in computer networks is the Precision Time Protocol, facilitating the exchange of synchronization information among disparate entities to ensure temporal alignment. While communication networks inherently introduce random delays, such as packet delay variation, the protocol does not dive deeply into the details of time estimation based on these measurements. Existing literature offers numerous methods to contend with uncertainty in measurements for clock state estimation, with those employing Kalman filtering exhibiting superior performance and moderate computational complexity. This dissertation makes a distinctive contribution by proposing adaptive algorithms for the filtering process to manage unknown packet delay variation.

Chapter 3 proposes a framing technique to enhance the effective data rate of UWB communication, leading to shorter measurement intervals and more accurate estimation. However, this comes at the cost of increased UWB packet size, resulting in higher ranging error. Nevertheless, the dissertation concludes that this increment is negligible compared to errors arising from propagation delay variation. In contrast, Chapter 4 explores the feasibility of synchronizing two adjustable PTP clocks via UWB communication. The chapter presents an implementation and evaluation of the problem, highlighting the challenge posed by the local clock domain change.

Chapter 5 presents theoretically optimal frequency drift estimation (in a maximum likelihood sense) in UWB networks, employing parameter estimation in processes with latent states through an Expectation-Maximization technique. While this method is

optimal in a maximum-likelihood sense, it comes with significant computational complexity. Chapter 6 shifts the focus to the PTP domain, introducing an adaptive method based on the residual covariance fitting technique to address variable measurement noise. This method offers near-optimal estimation of both clock state and measurement noise, with a modest computational complexity. To tailor the Adaptive Kalman Filter to the problem, a new measurement model is introduced with a time-invariant measurement noise covariance matrix.

Bibliography

- [1] O. Seijo, J. A. Lopez-Fernández, H.-P. Bernhard, and I. Val. Enhanced Timestamping Method for Subnanosecond Time Synchronization in IEEE 802.11 Over WLAN Standard Conditions. *IEEE Transactions on Industrial Informatics*, 16(9):5792–5805, 2020.
- [2] Z. Idrees, J. Granados, Y. Sun, S. Latif, L. Gong, Z. Zou, and L. Zheng. IEEE 1588 for Clock Synchronization in Industrial IoT and Related Applications: A Review on Contributing Technologies, Protocols and Enhancement Methodologies. *IEEE Access*, 8:155660–155678, 2020.
- [3] M. Lipiński, T. Włostowski, J. Serrano, and P. Alvarez. White rabbit: a PTP application for robust sub-nanosecond synchronization. In *2011 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, pages 25–30, 2011.
- [4] L. Silva, P. Pedreiras, P. Fonseca, and L. Almeida. On the adequacy of SDN and TSN for Industry 4.0. In *2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC)*, pages 43–51, 2019.
- [5] V. Balasubramanian, M. Aloqaily, and M. Reisslein. An SDN architecture for time sensitive industrial IoT. *Computer Networks*, 186:107739, 2021.
- [6] Y. Zhang, J. Wu, M. Liu, and A. Tan. TSN-based routing and scheduling scheme for Industrial Internet of Things in underground mining. *Engineering Applications of Artificial Intelligence*, 115:105314, 2022.
- [7] M. Ashjaei, L. Lo Bello, M. Daneshtalab, G. Patti, S. Saponara, and S. Mubeen. Time-Sensitive Networking in automotive embedded systems: State of the art and research opportunities. *Journal of Systems Architecture*, 117:102137, 2021.
- [8] J. Prados-Garzon and T. Taleb. Asynchronous Time-Sensitive Networking for 5G Backhauling. *IEEE Network*, 35(2):144–151, 2021.
- [9] D. Qing, Z. Hongxiang, and Y. Lijun. Research on 5G High Precision Time Synchronous Networking Scheme. In *2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, pages 1020–1024, 2021.

- [10] J. Song, M. Kubomi, J. Zhao, and D. Takita. Time synchronization performance analysis considering the frequency offset inside 5G-TSN network. In *2021 17th International Symposium on Wireless Communication Systems (ISWCS)*, pages 1–6, 2021.
- [11] IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008)*, pages 1–499, 2020.
- [12] J. A. del Peral-Rosado, O. Renaudin, C. Gentner, R. Raulefs, E. Dominguez-Tijero, A. Fernandez-Cabezas, F. Blazquez-Luengo, G. Cueto-Felgueroso, A. Chassaigne, D. Bartlett, F. Grec, L. Ries, R. Prieto-Cerdeira, J. A. Lopez-Salcedo, and G. Seco-Granados. Physical-Layer Abstraction for Hybrid GNSS and 5G Positioning Evaluations. In *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, pages 1–6, 2019.
- [13] M. Martalò, S. Perri, G. Verdano, F. De Mola, F. Monica, and G. Ferrari. Improved UWB TDoA-Based Positioning Using a Single Hotspot for Industrial IoT Applications. *IEEE Transactions on Industrial Informatics*, 18(6):3915–3925, 2022.
- [14] IEEE Standard for Low-Rate Wireless Networks. *IEEE Std 802.15.4-2020 (Revision of IEEE Std 802.15.4-2015)*, pages 1–800, 2020.
- [15] J. Friedrich, J. Tiemann, and C. Wietfeld. Accurate Multi-Zone UWB TDOA Localization utilizing Cascaded Wireless Clock Synchronization. In *2021 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, 2021.
- [16] Y. Cheng and T. Zhou. UWB Indoor Positioning Algorithm Based on TDOA Technology. In *2019 10th International Conference on Information Technology in Medicine and Education (ITME)*, pages 777–782, 2019.
- [17] F. Bonafini, P. Ferrari, A. Flammini, S. Rinaldi, and E. Sisinni. Exploiting Time Synchronization as Side Effect in UWB Real-Time Localization Devices. In *2018 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*, pages 1–6, 2018.
- [18] M. Raza, N. Aslam, H. Le-Minh, S. Hussain, Y. Cao, and N. M. Khan. A Critical Analysis of Research Potential, Challenges, and Future Directives in Industrial Wireless Sensor Networks. *IEEE Communications Surveys Tutorials*, 20(1):39–95, 2018.
- [19] I. Guvenc and C.-C. Chong. A Survey on TOA Based Wireless Localization and NLOS Mitigation Techniques. *IEEE Communications Surveys Tutorials*, 11(3):107–124, 2009.
- [20] S. Gezici and H. V. Poor. Position Estimation via Ultra-Wide-Band Signals. *Proceedings of the IEEE*, 97(2):386–403, 2009.

- [21] W. Chantaweksomboon, C. Suwatthikul, S. Manatrinon, K. Athikulwongse, K. Kaemarungsi, R. Ranron, and P. Suksompong. On performance study of UWB real time locating system. In *2016 7th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)*, pages 19–24, 2016.
- [22] GPS.gov: Interface Control Documents — gps.gov. <https://www.gps.gov/technical/icwg/>. [Accessed 22-01-2025].
- [23] J. Martin, J. Burbank, W. Kasch, and P. D. L. Mills. Network Time Protocol Version 4: Protocol and Algorithms Specification. RFC 5905, June 2010.
- [24] IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std 1588-2002*, pages 1–154, 2002.
- [25] 3rd Generation Partnership Project (3GPP). ETSI TS 122 104 V18.4.0 (2024-07): 5G; Service requirements for cyber-physical control applications in vertical domains, 2024.
- [26] Audio Engineering Society (AES). AES67-2018 - AES standard for audio applications of networks - High-performance streaming audio-over-IP interoperability, 2018.
- [27] IEEE Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications. *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pages 1–421, 2020.
- [28] IEEE Standard Profile for Use of IEEE 1588 Precision Time Protocol in Power System Applications, 2011.
- [29] E. Securities and M. A. (ESMA). MiFID II/MiFIR: Regulatory Technical and Implementing Standards. Technical report, ESMA, 2017.
- [30] IEEE Standards Association. IEEE 1588 - Precision Time Protocol (PTP) Power Profile, 2008.
- [31] X. Chen, Y. Li, and S. Wang. 5G Time Synchronization for Smart Factories. *arXiv preprint*, 2022.
- [32] Timing characteristics of synchronous Ethernet equipment slave clock (EEC), 2018.
- [33] Y. Shen, Z. Ji, C. Ma, and M. Gao. A Bayesian Detect to Track System for Robust Visual Object Tracking and Semi-Supervised Model Learning. *arXiv preprint arXiv:2205.02371*, 2022.
- [34] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo. Fast Image Recovery Using Variable Splitting and Constrained Optimization. *IEEE Transactions on Image Processing*, 19(9):2345–2356, 2010.
- [35] S. M. Kay. *Fundamentals of Statistical Signal Processing, Volume III: Practical Algorithm Development*. Prentice Hall, 2013.

- [36] A. Haug. A Tutorial on Bayesian Estimation and Tracking Techniques Applicable to Nonlinear and Non-Gaussian Processes. Technical report, The MITRE Corporation, 2005.
- [37] M. A. T. Figueiredo. A Review of Bayesian Machine Learning Principles, Methods, and Applications. *International Journal of Innovative Science and Research Technology*, 8(5):2427–2435, 2023.
- [38] S. Särkkä. *Bayesian Filtering and Smoothing*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2013.
- [39] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960.
- [40] H. E. Rauch, F. Tung, and C. T. Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, 1965.
- [41] A. Pásztor and D. Veitch. PC based precision timing without GPS. *SIGMETRICS Perform. Eval. Rev.*, 30(1):1–10, jun 2002.
- [42] 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*, 52(2):289–296, 2005.
- [43] T. Ishizaki, T. Ichimura, T. Kawaguchi, Y. Yano, and Y. Hanado. Higher-Order Allan Variance for Atomic Clocks of Arbitrary Order: Mathematical Foundation. 2023.
- [44] W. Riley and D. Howe. Handbook of Frequency Stability Analysis, 2008-07-01 00:07:00 2008.
- [45] Tcxo facts — quartzpro.com. <https://www.quartzpro.com/tcxofacts.html>. [Accessed 06-08-2024].
- [46] M. Kihara, S. Ono, and P. Eskelinen. *Digital Clocks for Synchronization and Communications*. Artech House telecommunications library. Artech House, 2006.
- [47] IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pages 1–269, 2008.
- [48] IEEE Standard for Local and Metropolitan Area Networks—Timing and Synchronization for Time-Sensitive Applications. *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)*, pages 1–421, 2020.
- [49] P. Wolfrum, R. L. Scheiterer, and D. Obradovic. An optimal control approach to clock synchronization. In *2010 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, pages 122–128, 2010.

- [50] C. Na, P. Wolfrum, D. Obradovic, and R. L. Scheiterer. Optimal estimation and control of clock synchronization following the Precision Time Protocol. In *2010 IEEE International Conference on Control Applications*, pages 1767–1772, 2010.
- [51] K. J. Astrom and R. M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, USA, 2008.
- [52] C. Novaes, I. Freire, A. Klautau, I. Almeida, and E. Medeiros. Analysis of Kalman Filtering for Clock Synchronization in PTP-Unaware Networks. In *2021 IEEE Latin-American Conference on Communications (LATINCOM)*, pages 1–6, 2021.
- [53] A. Althoubi, R. Alshahrani, and H. Peyravi. Delay Analysis in IoT Sensor Networks. *Sensors*, 21(11), 2021.
- [54] H. Puttnies, P. Danielis, A. R. Sharif, and D. Timmermann. Estimators for Time Synchronization—Survey, Analysis, and Outlook. *IoT*, 1(2):398–435, 2020.
- [55] I. Freire, C. Novaes, I. Almeida, E. Medeiros, M. Berg, and A. Klautau. Clock Synchronization Algorithms Over PTP-Unaware Networks: Reproducible Comparison Using an FPGA Testbed. *IEEE Access*, 9:20575–20601, 2021.
- [56] A. Guruswamy, R. S. Blum, S. Kishore, and M. Bordoyna. Minimax Optimum Estimators for Phase Synchronization in IEEE 1588. *IEEE Transactions on Communications*, 63(9):3350–3362, 2015.
- [57] G. Cena, S. Scanzio, and A. Valenzano. Reliable comparison of clock discipline algorithms for time synchronization protocols. In *2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA)*, pages 1–9, 2015.
- [58] H. Puttnies, P. Danielis, and D. Timmermann. PTP-LP: Using Linear Programming to Increase the Delay Robustness of IEEE 1588 PTP. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–7, 2018.
- [59] A. K. Karthik and R. S. Blum. Robust Clock Skew and Offset Estimation for IEEE 1588 in the Presence of Unexpected Deterministic Path Delay Asymmetries. *IEEE Transactions on Communications*, 68(8):5102–5119, 2020.
- [60] R. Exel and F. Ring. Improved clock synchronization accuracy through optimized servo parametrization. In *2013 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS) Proceedings*, pages 65–70, 2013.
- [61] F. Ring, T. Bigler, and R. Exel. Synchronization robustness using Kalman-based clock servos. In *2015 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*, pages 64–69, 2015.
- [62] G. Giorgi and C. Narduzzi. Performance Analysis of Kalman-Filter-Based Clock Synchronization in IEEE 1588 Networks. *IEEE Transactions on Instrumentation and Measurement*, 60(8):2902–2909, 2011.

- [63] G. Giorgi. An Event-Based Kalman Filter for Clock Synchronization. *IEEE Transactions on Instrumentation and Measurement*, 64(2):449–457, 2015.
- [64] G. Giorgi and C. Narduzzi. A resilient Kalman filter based servo clock. In *2013 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS) Proceedings*, pages 59–64, 2013.
- [65] X. Liu and H. Wang. Reliability-Aware Clock Offset Tracking With Timestamp Validation in IEEE 1588 Networks. *IEEE Communications Letters*, pages 1–1, 2024.
- [66] S. Shirmohammadi, L. Mari, and D. Petri. On the Commonly-Used Incorrect Visual Representation of Accuracy and Precision. *IEEE Instrumentation & Measurement Magazine*, 24(1):45–49, 2021.
- [67] D. Coppens, E. De Poorter, A. Shahid, S. Lemey, and C. Marshall. An Overview of Ultra-WideBand (UWB) Standards(IEEE 802.15.4, FiRa, Apple): Interoperability Aspects and Future Research Directions, 2022.
- [68] P. Mayer, M. Magno, C. Schnetzler, and L. Benini. EmbedUWB: Low Power Embedded High-Precision and Low Latency UWB Localization. In *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, pages 519–523, 2019.
- [69] B. Silva, Z. Pang, J. Åkerberg, J. Neander, and G. Hancke. Experimental study of UWB-based high precision localization for industrial applications. In *2014 IEEE International Conference on Ultra-WideBand (ICUWB)*, pages 280–285, 2014.
- [70] J. Pan. Medical applications of ultra-wideband (uwb). *Survey Paper*, 2007.
- [71] H.-L. Bloecher, A. Sailer, G. Rollmann, and J. Dickmann. 79 GHz uwb automotive short range radar—spectrum allocation and technology trends. *Advances in Radio Science*, 7(B.3):61–65, 2009.
- [72] G. Oguntala, R. Abd-Alhameed, S. Jones, J. Noras, M. Patwary, and J. Rodriguez. Indoor location identification technologies for real-time IoT-based applications: An inclusive survey. *Computer Science Review*, 30:55–79, 2018.
- [73] M. M. Schack, M. Jacob, and T. Kierner. Comparison of in-car uwb and 60 ghz channel measurements. In *Proceedings of the Fourth European Conference on Antennas and Propagation (EuCAP) 2010*, pages 1–5, 2010.
- [74] J. Foerster, E. Green, S. Somayazulu, and D. Leeper. Ultra-wideband technology for short-or medium-range wireless communications. *Intel technology journal*, 2001.
- [75] D. Neiryneck, E. Luk, and M. McLaughlin. An alternative double-sided two-way ranging method. In *2016 13th Workshop on Positioning, Navigation and Communications (WPNC)*, pages 1–4, 2016.

- [76] S. Gezici, Z. Tian, G. Giannakis, H. Kobayashi, A. Molisch, H. Poor, and Z. Sahinoglu. Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks. *IEEE Signal Processing Magazine*, 22(4):70–84, 2005.
- [77] Z. Sahinoglu, S. Gezici, and I. Güvenc. *Ultra-wideband Positioning Systems: Theoretical Limits, Ranging Algorithms, and Protocols*. Cambridge University Press, 2008.
- [78] A. Poulou, Z. Emersic, O. Steven Eyobu, and D. Seog Han. An Accurate Indoor User Position Estimator For Multiple Anchor UWB Localization. In *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 478–482, 2020.
- [79] A. Poulou and D. S. Han. Feature-Based Deep LSTM Network for Indoor Localization Using UWB Measurements. In *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIIC)*, pages 298–301, 2021.
- [80] J. Cholz, A. Hernandez-Solana, and A. Valdovinos. Strategies for optimizing latency and resource utilization in multiple target UWB-based tracking. In *2011 IEEE Wireless Communications and Networking Conference*, pages 838–843, 2011.
- [81] D. Ltd. *DW1000 User Manual, "How To use, configure and program the DW1000 UWB transmitter"*. Qorvo, 2017.
- [82] Decawave. *DWM1001 System Overview And Performance*, version 2.0 edition, 2018.
- [83] DWM1001C - Qorvo — qorvo.com. <https://www.qorvo.com/products/p/DWM1001C>. [Accessed 07-08-2024].
- [84] I. Val, O. Seijo, R. Torrego, and A. Astarloa. IEEE 802.1AS Clock Synchronization Performance Evaluation of an Integrated Wired–Wireless TSN Architecture. *IEEE Transactions on Industrial Informatics*, 18(5):2986–2999, 2022.
- [85] H. Shi, A. Aijaz, and N. Jiang. Evaluating the Performance of Over-the-Air Time Synchronization for 5G and TSN Integration. In *2021 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pages 1–6, 2021.
- [86] 3rd Generation Partnership Project (3GPP). 3GPP TS 23.501: System Architecture for the 5G System (Release 16). Technical report, 3rd Generation Partnership Project (3GPP), 2020. Release 16.
- [87] T. Adame, M. Carrascosa-Zamacois, and B. Bellalta. Time-Sensitive Networking in IEEE 802.11be: On the Way to Low-Latency WiFi 7. *Sensors*, 21(15), 2021.
- [88] J. Haxhibeqiri, X. Jiao, M. Aslam, I. Moerman, and J. Hoebeke. Enabling TSN over IEEE 802.11: Low-overhead Time Synchronization for Wi-Fi Clients. In *2021 22nd IEEE International Conference on Industrial Technology (ICIT)*, volume 1, pages 1068–1073, 2021.

- [89] IEEE Standard for Information technology—Telecommunications and information exchange between systems Local and metropolitan area networks—Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std 802.11-2016 (Revision of IEEE Std 802.11-2012)*, pages 1–3534, 2016.
- [90] A. Jiménez and F. Seco. Finding objects using UWB or BLE localization technology: A museum-like use case. In *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–8, 2017.
- [91] W. Shule, C. M. Almansa, J. P. Queralt, Z. Zou, and T. Westerlund. UWB-Based Localization for Multi-UAV Systems and Collaborative Heterogeneous Multi-Robot Systems. *Procedia Computer Science*, 175:357–364, 2020. MobiSPC.
- [92] M. Drobczyk, C. Strowik, and C. Philpot. A Wireless Communication and Positioning Experiment for the ISS Based on IR-UWB. In *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, 2017.
- [93] S. Rinaldi, A. Musatti, A. Depari, P. Ferrari, A. Flammini, and E. Sisinni. An Experimental Characterization of Chain of PLLs for Wired Clock Synchronization of UWB Anchors for Indoor Location. In *2022 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pages 1–6, 2022.
- [94] R. Zou, N. Wu, Z. Qu, and J. Chen. Design and Implementation of a High-precision Wireless Clock Synchronization System Based on UWB. In *2022 4th International Conference on Intelligent Control, Measurement and Signal Processing (ICMSP)*, pages 1094–1099, 2022.
- [95] P. Ferrari, P. Bellagente, A. Depari, A. Flammini, M. Pasetti, S. Rinaldi, and E. Sisinni. Resilient Time Synchronization Opportunistically Exploiting UWB RTLS Infrastructure. *IEEE Transactions on Instrumentation and Measurement*, 71:1–10, 2022.
- [96] M. Segura, S. Niranjayan, H. Hashemi, and A. F. Molisch. Experimental demonstration of nanosecond-accuracy wireless network synchronization. In *2015 IEEE International Conference on Communications (ICC)*, pages 6205–6210, 2015.
- [97] B. Xue, Z. Li, P. Lei, Y. Wang, and X. Zou. Wicsync: A wireless multi-node clock synchronization solution based on optimized UWB two-way clock synchronization protocol. *Measurement*, 183:109760, 2021.
- [98] ARM Limited. *Cortex-M4 Processor Technical Reference Manual*, 2010. ARM Cortex-M4 is a high-performance processor designed for low-power applications.
- [99] ST. *Reference manual, STM32F405/415, STM32F407/417, STM32F427/437 and STM32F429/439 advanced Arm-based 32-bit MCUs*, 2021.

- [100] K. Correll, N. A. Barendt, and M. S. Branicky. Design Considerations for Software Only Implementations of the IEEE 1588 Precision Time Protocol. 2005.
- [101] S. Microelectronics. RM0090 Reference Manual. https://www.st.com/resource/en/reference_manual/. [Accessed 13-08-2024].
- [102] 3GPP. Service requirements for cyber-physical control applications in vertical domains. Technical Specification (TS) 22.104, 3rd Generation Partnership Project (3GPP), June 2024.
- [103] B. Choi, K. La, and S. Lee. UWB TDOA/TOA measurement system with wireless time synchronization and simultaneous tag and anchor positioning. In *2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*, pages 1–6, 2018.
- [104] Q. Shi, S. Zhao, X. Cui, M. Lu, and M. Jia. Anchor self-localization algorithm based on UWB ranging and inertial measurements. *Tsinghua Science and Technology*, 24(6):728–737, 2019.
- [105] M. Pelka and H. Hellbrück. S-TDoA — Sequential time difference of arrival — A scalable and synchronization free approach for Positioning. In *2016 IEEE Wireless Communications and Networking Conference*, pages 1–6, 2016.
- [106] J. Kulmer, S. Hinteregger, B. Großwindhager, M. Rath, M. S. Bakr, E. Leitinger, and K. Witrisal. Using DecaWave UWB transceivers for high-accuracy multipath-assisted indoor positioning. In *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 1239–1245, 2017.
- [107] S. Shah and T. Demechai. Multiple Simultaneous Ranging in IR-UWB Networks. *Sensors*, 19(24), 2019.
- [108] N. M. Senevirathna, O. De Silva, G. K. I. Mann, and R. G. Gosine. Kalman Filter based Range Estimation and Clock Synchronization for Ultra Wide Band Networks. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9027–9032, 2020.
- [109] Y. Qi, H. Suda, and H. Kobayashi. On time-of-arrival positioning in a multipath environment. In *IEEE 60th Vehicular Technology Conference, 2004. VTC2004-Fall. 2004*, volume 5, pages 3540–3544 Vol. 5, 2004.
- [110] Y. Qi and R. Kohno. Mitigation of sampling-induced errors in delay estimation. In *2005 IEEE International Conference on Ultra-Wideband*, pages 6 pp.–, 2005.
- [111] A. F. Molisch, K. Balakrishnan, D. Cassioli, C.-C. Chong, S. Emami, A. Fort, Johan, Karedal, J. Kunisch, H. G. Schantz, U. G. Schuster, and K. Siwiak. IEEE 802.15.4a channel model-final report. 2004.

- [112] A. Saleh and R. Valenzuela. A Statistical Model for Indoor Multipath Propagation. *IEEE Journal on Selected Areas in Communications*, 5(2):128–137, 1987.
- [113] B. Sklar. Rayleigh fading channels in mobile digital communication systems .I. Characterization. *IEEE Communications Magazine*, 35(7):90–100, 1997.
- [114] A. F. Molisch, D. Cassioli, C.-C. Chong, S. Emami, A. Fort, B. Kannan, J. Karedal, J. Kunisch, H. G. Schantz, K. Siwiak, and M. Z. Win. A Comprehensive Standardized Model for Ultrawideband Propagation Channels. *IEEE Transactions on Antennas and Propagation*, 54(11):3151–3166, 2006.
- [115] M. NAKAGAMI. The m-Distribution—A General Formula of Intensity Distribution of Rapid Fading. In W. HOFFMAN, editor, *Statistical Methods in Radio Wave Propagation*, pages 3–36. Pergamon, 1960.
- [116] Decawave. Non line of sight operation and optimizations to improve performance in DW1000 based systems, 2014.
- [117] A. Bletsas. Evaluation of Kalman filtering for network time keeping. *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on*, 52:1452 – 1460, 10 2005.
- [118] Y. Xiao, H. Wang, and P. Liu. Nonlinear wireless time synchronization based on variational Bayesian adaptive filtering. In *2021 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pages 1–6, 2021.
- [119] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.
- [120] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, Ltd, May 2000.
- [121] M. Hamer and R. D’Andrea. Self-Calibrating Ultra-Wideband Network Supporting Multi-Robot Localization. *IEEE Access*, 6:22292–22304, 2018.
- [122] N. F. TODA, F. H. SCHLEE, and P. OBSHARSKY. Region of Kalman filter convergence for several autonomous navigation modes. *AIAA Journal*, 7(4):622–627, April 1969.
- [123] Y. Ha, E. Pak, J. Park, T. Kim, and J. W. Yoon. Clock Offset Estimation for Systems With Asymmetric Packet Delays. *IEEE/ACM Transactions on Networking*, 31(4):1838–1853, 2023.
- [124] M. Khodarahmi and V. Maihami. A Review on Kalman Filter Models. *Archives of Computational Methods in Engineering*, 30(1):727–747, October 2022.

- [125] W. Youn, N. Y. Ko, S. A. Gadsden, and H. Myung. A Novel Multiple-Model Adaptive Kalman Filter for an Unknown Measurement Loss Probability. *IEEE Transactions on Instrumentation and Measurement*, 70:1–11, 2021.
- [126] M. Karasalo and X. Hu. An optimization approach to adaptive Kalman filtering. *Automatica*, 47(8):1785–1793, 2011.
- [127] P. Bania and J. Baranowski. Field Kalman Filter and its approximation. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 2875–2880, 2016.
- [128] Z. Durovic and B. Kovacevic. Robust estimation with unknown noise statistics. *IEEE Transactions on Automatic Control*, 44(6):1292–1296, 1999.
- [129] A. H. Mohamed and K. P. Schwarz. Adaptive Kalman Filtering for INS/GPS. *Journal of Geodesy*, 73(4):193–203, May 1999.
- [130] V. A. Bavdekar, A. P. Deshpande, and S. C. Patwardhan. Identification of process and measurement noise covariance for state and parameter estimation using extended Kalman filter. *Journal of Process Control*, 21(4):585–601, 2011.
- [131] R. Exel, T. Bigler, and T. Sauter. Asymmetry Mitigation in IEEE 802.3 Ethernet for High-Accuracy Clock Synchronization. *IEEE Transactions on Instrumentation and Measurement*, 63(3):729–736, 2014.
- [132] H. Abubakari and S. Sastry. IEEE 1588 style synchronization over wireless link. In *2008 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, pages 127–130, 2008.
- [133] C. Montella. The Kalman Filter and Related Algorithms: A Literature Review. 05 2011.
- [134] GitHub - ptpd/ptpd: PTPd official source - master branch a.k.a. trunk — github.com. <https://github.com/ptpd/ptpd>. [Accessed 05-09-2024].
- [135] The Linux PTP Project — linuxptp.sourceforge.net. <https://linuxptp.sourceforge.net/>. [Accessed 05-09-2024].
- [136] I. Greenberg, N. Yannay, and S. Mannor. Optimization or Architecture: How to Hack Kalman Filtering. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 50482–50505. Curran Associates, Inc., 2023.
- [137] P. Chauchat, A. Bellés, D. Medina, and J. Vilà-Valls. Insights on Adaptive Robust Filtering for Navigation Under Harsh Time-Varying Environments. In *2024 32nd European Signal Processing Conference (EUSIPCO)*, pages 1227–1231, 2024.
- [138] S. Zhong, Z. Wang, G. Wang, Y. Zhou, X. Zhou, and B. Peng. Robust adaptive filtering based on M-estimation-based minimum error entropy criterion. *Information Sciences*, 658:120026, 2024.

- [139] X. Liu and H. Wang. Robust Clock Parameters Tracking for IEEE 1588 With Asymmetric Packet Delays in Industrial Networks. *IEEE Transactions on Communications*, pages 1–1, 2024.

Publications

Journal papers

- [J1] G. Hollósi and I. Moldován. Ultra Wideband-based wireless synchronization of IEEE 1588 clocks. *Infocommunications Journal*, 15(2):21–28, 2023.
- [J2] G. Hollósi and D. Ficzere. Adaptive Kalman Filtering in Offset Estimation for Precision Time Protocol. *IEEE Transactions on Industrial Informatics*, 21(1):396–404, 2025.
- [J3] G. Hollósi, C. Lukovszki, I. Moldován, S. Plósz, and F. Harasztos. Monocular indoor localization techniques for smartphones. *Acta Universitatis Sapientiae, Informatica*, 8(2):186–215, 2016.
- [J4] G. Hollósi, C. Lukovszki, M. Bancsics, and G. Magyar. Traffic Swarm Behaviour: Machine Learning and Game Theory in Behaviour Analysis. *Infocommunications Journal*, 13(4):19–27, 2021.
- [J5] G. Hollósi, C. Lukovszki, I. Moldován, S. Plósz, and F. Harasztos. Survey on Monocular Odometry for Conventional Smartphones. *Infocommunications Journal*, page 25, 2016.
- [J6] A. Frankó, G. Hollósi, D. Ficzere, and P. Varga. Applied Machine Learning for IIoT and Smart Production and Methods to Improve Production Quality, Safety and Sustainability. *Sensors*, 22(23), 2022.

Conference proceedings

- [C1] G. Hollósi, C. Lukovszki, and M. Bancsics. Radio Resource Efficient UWB Measurement System Design and Performance Analysis for TWR-based Ranging. In *2022 IEEE 5th International Conference on Industrial Cyber-Physical Systems (ICPS)*, pages 1–6, 2022.
- [C2] G. Hollósi. Distribution of ultra wideband (UWB) receive timestamps in dense indoor environment based on the Saleh-Valenzuela channel model. In *2022 14th International Conference on Communications (COMM)*, pages 1–5, 2022.

- [C3] G. Hollósi. Bayesian Measurement Noise Estimation in Ultra Wide Band Systems for Clock Synchronization. In *2023 IEEE 19th International Conference on Factory Communication Systems (WFCS)*, pages 1–8, 2023.
- [C4] S. Plósz, Z. Kertész, C. Lukovszki, D. Kovács, I. Moldován, and G. Hollósi. Practical aspects of visual recognition for indoor mobile positioning. In *2013 IEEE 4th International Conference on Cognitive Infocommunications (CogInfoCom)*, pages 527–532, 2013.
- [C5] S. Plósz, C. Lukovszki, and G. Hollósi. Visual feature recognition based indoor localization. In *1st International Conference and Exhibition on Future RFID Technologies*, pages 143–150. Eszterházy Károly Főiskola, 11 2014.
- [C6] D. Ficzer, G. Hollósi, A. Frankó, and A. Gulyás. Random walk for generalization in goal-directed human navigation on Wikipedia. In *The 11th International Conference on Complex Networks and their Applications*, 06 2022.
- [C7] A. Frankó and G. Hollósi. Settling Issues in IEEE 802.1AS Networks in PI Based Clock Servos. In *2023 19th International Conference on Network and Service Management (CNSM)*, pages 1–7, 2023.
- [C8] D. Ficzer, G. Hollósi, A. Frankó, and A. Gulyás. Random Walk for Generalization in Goal-Directed Human Navigation on Wikipedia. In H. Cherifi, R. N. Mantegna, L. M. Rocha, C. Cherifi, and S. Micciché, editors, *Complex Networks and Their Applications XI*, pages 202–213, Cham, 2023. Springer International Publishing.
- [C9] D. Ficzer, G. Hollósi, and P. Varga. Live traffic analysis on S1-MME interface using LSTM autoencoder. In *2023 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, pages 68–73, 2023.
- [C10] D. Ficzer, G. Hollósi, A. Frankó, and P. Varga. AI Toolbox Concept for the Arrowhead Framework. In *2023 19th International Conference on Network and Service Management (CNSM)*, pages 1–7, 2023.
- [C11] D. Ficzer, G. Hollósi, A. Frankó, P. Varga, and J. Biró. Orderliness of Navigation Patterns in Hyperbolic Complex Networks. In H. Cherifi, L. M. Rocha, C. Cherifi, and M. Donduran, editors, *Complex Networks & Their Applications XII*, pages 271–282, Cham, 2024. Springer Nature Switzerland.
- [C12] D. Ficzer, G. Hollósi, A. Frankó, P. Varga, and J. Biró. The Traveling Salesman Problem on the Hyperbolic Plane. In *2023 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 433–437, 2023.
- [C13] G. Hollósi, D. Ficzer, A. Frankó, M. Bancsics, R. AlMahasneh, C. Lukovszki, and P. Varga. AIMS5.0 AI Toolbox: Enabling Efficient Knowledge Sharing for Industrial AI. In *NOMS 2024-2024 IEEE Network Operations and Management Symposium*, pages 1–6, 2024.

- [C14] G. Hollósi, D. Ficzere, and P. Varga. Generative AI for Low-Level NETCONF Configuration in Network Management Based on YANG Models. In *2024 20th International Conference on Network and Service Management (CNSM)*, pages 1–7, 2024.
- [C15] R. AlMahasneh, G. Hollósi, D. Ficzere, M. Bancsics, C. Lukovszki, and P. Varga. Uncovering Common AI Challenges Across Industrial Domains in the Transition to Industry 5.0. In *2024 20th International Conference on Network and Service Management (CNSM)*, pages 1–7, 2024.