FLock: Enabling Self-Localisation of Embedded Devices in Mobile and Multi-Hop UWB Systems

Maximilian Schuh
Institute of Technical Informatics
Graz University of Technology
Graz, Austria
schuh@tugraz.at

Markus Schuss
Institute of Technical Informatics
Graz University of Technology
Graz, Austria
markus.schuss@tugraz.at

Michael Baddeley
Secure Systems Research Center
Technology Innovation Institute
Abu Dhabi, United Arab Emirates
Michael.Baddeley@tii.ae

Kay Römer
Institute of Technical Informatics
Graz University of Technology
Graz, Austria
roemer@tugraz.at

Carlo Alberto Boano
Institute of Technical Informatics
Graz University of Technology
Graz, Austria
cboano@tugraz.at

Abstract

Precise localisation is a key enabler for mobile applications, yet it remains challenging in global navigation satellite system (GNSS)-denied environments such as buildings, tunnels, and caves. Ultra-wideband (UWB) has emerged as a promising technology for such scenarios, but existing UWB-based real-time localisation systems typically rely on fixed infrastructure and are ill-suited for rapidly deployed, mobile, multi-hop networks. This paper presents FLock, an infrastructure-free, **fl**ooding-assisted self-**loc**alisation system designed for resource-constrained devices. FLock addresses three core challenges: (i) efficiently gathering accurate distance measurements at scale via a one-to-many UWB ranging protocol, (ii) reliably disseminating distance information in highly mobile, multi-hop topologies using a protocol built on top of synchronous narrowband flooding, and (iii) performing on-device localisation through a lightweight gradient-descent-based engine. By tightly integrating these components, FLock provides both relative and absolute localisation. We validate FLock through simulations and real-world experiments with up to 30 UWB devices. The results show that FLock enables real-time localisation directly on embedded devices in both static and mobile scenarios, supporting practical deployment in demanding mobile environments while maintaining a localisation error below 36 cm (MAE) and an update rate of up to 3 Hz.

CCS Concepts

• **Networks** → **Location based services**; **Mobile ad hoc networks**; • **Computer systems organization** → *Sensor networks*; *Embedded software.*

Keywords

Ultra-Wideband, self-localisation, localisation networks, multi-hop localisation

1 Introduction

In recent years, precise location awareness has become a critical capability for mobile applications. This need becomes particularly evident in environments where GNSSs are unavailable, such as buildings, tunnels, or caves, where radio-based localisation techniques have proven indispensable. Building on this, the emergence and standardisation of UWB technology in the IEEE 802.15.4 standard, coupled with its excellent ranging performance, has driven the development of UWB-based real-time location systems (RTLSs) in recent years.

However, most of these systems require pre-installed infrastructure, such as static anchors with accurately known positions, which makes deployment and installation tedious and labour-intensive [5]. Furthermore, there are use cases that require an immediate, ad hoc deployment of multi-hop localisation systems and so-called *self-localisation* capabilities. Examples can be found in the domain of relative swarm localization [3, 29, 35] or the localisation of first-responders in unknown buildings [6, 31, 36]. We identify three main challenges for mobile self-localisation systems:

I. Efficient gathering of distance information. To self-localise a network, nodes must collect sufficient distance data from one another. UWB-based two-way ranging (TWR) schemes are a promising candidate for this task due to their accuracy. Intuitively, increasing the number of measured distances between neighbouring nodes enhances the network's self-localisation accuracy. However, this approach introduces a scalability challenge: in a fully connected network, the number of distance measurements grows quadratically with the number of nodes, making sequential pairwise distance measurements among all devices infeasible. Furthermore, given the number of distance measurements and the high update rate required in mobile scenarios, system-wide coordination is necessary to avoid collisions in the contested radio frequency (RF) domain. Therefore, systems are needed that enable the efficient collection of distance measurements across an increasing number of devices.

II. Sharing distance information in a mobile multi-hop network. Assuming that distance information has been gathered on individual nodes throughout the network, the next step is to combine this distributed information to enable self-localisation. To this end, UWB radios are often accompanied by secondary narrowband radios for communication and management e.g., within the FIRA standard [41] or the narrowband-assisted multi-millisecond scheme in the upcoming IEEE 802.15.4ab standard [17]. However, these solutions typically focus on one-to-one ranging and do not consider the dissemination of ranging information within large-scale multi-hop networks.

Additionally, while data collection has been well studied in the context of wireless sensor networks (WSNs), high mobility within the network poses a severe challenge, as it prevents the establishment of static, known communication routes. However, for the application of self-localisation in mobile, infrastructure-free, multi-hop networks, information must be shared reliably and promptly to support real-time operation and minimise transmission latency.

III. On-device self-localisation of a multi-hop UWB network. Once all distance information has been gathered, the positions have to be estimated. Several algorithms, including multilateration [4, 6, 38] and optimisation-based methods [5, 9, 16, 28], have been proposed to address this problem. Many existing methods are limited to small-scale (i.e., less than ten nodes), single-hop setups or are too computationally expensive and have only run on powerful edge devices. However, the scaling of ranging information within the network means that offloading computations to edge devices incurs high latency costs. This highlights the need for lightweight, scalable algorithms suitable for resource-constrained devices, such as Nordic's popular nRF52 series microcontroller units (MCUs) with a 64 MHz Cortex-M4.

Lack of a holistic approach. In this context, a large body of work investigated different aspects of the self-localisation of UWB networks [27]. Nevertheless, existing solutions from both industry and academia typically focus on small-scale, single-hop environments or address the complexity of managing multi-hop networks and of live data collection by relying on simulations or prerecorded data rather than real deployments. As a result, systems are required that address all the aforementioned challenges holistically.

Our contributions. In response to these demands, this paper presents FLocк, a **fl**ooding-assisted self-**loc**alisation system designed for mobile, constrained devices. FLocк integrates three main components: (i) an UWB-based ranging protocol for efficient one-to-many distance measurements among neighbouring nodes, (ii) a lightweight data dissemination protocol based on the Open Synchronous Flooding (OSF) framework that enables robust data sharing in high-mobility, multi-hop networks, and (iii) an *on-device* localisation engine employing a lightweight gradient descent algorithm. By integrating these components, FLocк provides both relative and absolute self-localisation capabilities without relying on fixed anchors or external infrastructure. To validate FLocк, we conduct simulations in diverse setups, compare localisation performance against other algorithms, and validate our results in real-world experiments with up to 29 nodes in a testbed environment. Our results demonstrate that FLocк delivers accurate (mean absolute error $\leq$ 36 cm), real-time (update rate $\leq$ 1 Hz) localisation directly on embedded devices in both static and mobile environments. FLocк's source code will be open-sourced to allow future development and extensions [33].

Paper structure. The remainder of this paper is structured as follows: After reviewing the related work in Section 2, we give an introduction to UWB and OSF in Section 3 before introducing the design and implementation of FLocк in Section 4. Section 5, investigates the system's performance in real-world scenarios covering static and mobile real-world experiments. Finally, Section 6 provides an outlook for future work before we conclude the paper in Section 7.

2 Related Work

In this section, we review self-localisation in sensor networks, focusing on recent UWB-based approaches summarised in Table 1.

Radio-based self-localisation in sensor networks. RF-based localisation of (mobile) nodes has been studied since the early days of WSNs. Due to the high complexity and deployment cost of pre-installed infrastructure – i.e., anchor nodes with known positions – or applications that demand infrastructure-free operations, the issues of self-calibration and collaborative localisation have quickly gained importance [2, 10, 24, 40]. These terms – self-calibration, self-localisation, self-positioning, and collaborative localisation – are often used interchangeably [27], referring to localisation of static anchors with partially known positions or relative positioning in mobile networks (e.g., drone swarms). The rise of UWB and its precise ranging capabilities has spurred new research.

Self-localisation in single-hop UWB networks. Hamer et al. [16] proposed a self-calibration for static anchors in a single-hop network. Using beacon messages in tight synchronisation, the devices continuously exchange ranging information and share estimates of their positions. Each device computes its own position via gradient-based optimisation, using distances and positions from neighbours. With three known positions, the approach led to an root mean square error (RMSE) of 9.7 cm in an eight-node single-hop network. Ridolfi et al. [28] extended this with on-device machine learning to detect and mitigate non-line-of-sight (NLOS) effects, achieving an accuracy of 2.4 cm in a 4-node single-hop network.

Multidimensional scaling (MDS) refers to a group of functions that were originally intended to visualise the relation between any set of data points. In recent years, however, MDS has also emerged as a key method for relative [7] and absolute [8, 9, 22] self-localisation in UWB networks. MDS reduces a dissimilarity matrix (i.e. distances) to a lower-dimensional space (i.e. node coordinates). *Classical* MDS provides a closed-form solution, building upon an eigenvalue decomposition and hence requires a square distance matrix, i.e. single-hop, all-to-all distance measurements. In contrast, the more general *metric* MDS uses the so-called SMACOF (scaling by majorizing a complicated function) approach. Here, the coordinates are determined by iteratively minimising a majorization function, called stress. Di Franco et al. explored metric MDS variants that account for NLOS errors [7] and incorporated known anchor positions [8, 9]. Experiments with up to 9 nodes achieved an accuracy of 45 cm [9]. Koledoye et al. [22] handled missing range measurements by introducing weights and published their Python implementation [21]. Due to the complexity of metric MDS and the iterative stress-majorization optimisation often used, these methods have not been run on-device yet.

However, Santoro et al. [30] used classic MDS with time-series data to resolve ambiguities and improve relative localisation. They demonstrated a mobile node localising five static nodes in a one-hop setup. The closed-form classic MDS enabled on-device computation, achieving an RSME below 20 cm.

From single to multi-hop. Corbalán et al. [5] evaluated metric MDS for anchor self-localisation in large-scale multi-hop networks with prerecorded data of up to 36 nodes across three environments. In these deployments, MDS proved to be feasible with mean errors well below 50 cm.

Table 1: Summary of UWB-based self-localisation solutions.

Reference	mobility	#Nodes	Localisation algorithm	on-device	data-collection/ distribution	multi-hop capable	rel./abs. localisation
Hamer et al. [16]	×	8	distributed gradient-descent	✓	UWB	×	absolute
Ridolfi et al. [28]	×	4	distributed gradient-descent	✓	UWB	×	absolute
Di Franco et al. [9]	×	9	metric MDS / dynamic MDS	×	not-specified	×	absolute
Di Franco et al. [7]	✓	13	metric MDS	×	not-specified	×	relative
Santoro et al. [30]	×	6	classical MDS	✓	WiFi	×	relative
Cao et al. [4]	✓	12	multilateration	×	UWB	~ (2-hops)	relative
Dhekne et al. [6]	✓	10	multilateration	×	WiFi	~ (2-hops)	absolute
Corbalán et al. [5]	×	36	metric MDS	×	prerecorded data	✓	absolute
Herbruggen et al. [38]	×	18	multilateration + BFGS	×	(single-hop) sub-GHz RF	✓	absolute
FLocк	✓	30	centralized gradient-descent	✓	narrowband flooding	✓	both

Herbruggen et al. [38] proposed a different approach. They presented a three-stage multi-hop self-calibration algorithm for UWB anchors: First, multilateration for initial anchor estimates, second, an optimisation for position refinement and finally, a mobile tag can be used to refine uncertain anchor positions. In an Industry 4.0 testbed (18 anchors, metal racks), they achieved a mean absolute error of 21.6 cm.

Dhekne et al. [6] developed a localisation system for first responders. A mobile drone outside a building enabled the localisation of devices inside by leveraging time-series data. Devices without direct drone-connectivity estimate positions via two-hop localisation using a multilateration approach.

Cao et al. [4] introduced a scheduling algorithm for efficient ranging in dynamic UWB networks. They also addressed relative localisation within the network. Each node maps its local neighbourhood via trilateration and least-squares optimisation. By sharing the local map among the direct neighbours, an off-device algorithm merges these maps into larger ones. However, the localisation performance of this approach was not evaluated.

The gap to be filled. Extensive research has already advanced self-localisation of UWB networks (see Table 1). However, there is still no system that unites live data collection, in-network aggregation, and on-device real-time computation for multi-hop networks. Most work remains limited to single-hop networks [9, 16, 28]. Multi-hop studies often rely on pre-recorded data and overlook challenges like measurement coordination [5, 38], or support only a limited number of hops [3, 6]. Furthermore, on-device self-localisation remains restricted to small-scale single-hop setups [16, 28].

3 Background

This section provides an overview of the underlying technologies on which FLocк is built.

A primer on ultra-wideband. In contrast to narrowband, UWB signals are characterised by their large bandwidth (typically above 500 MHz). The most common UWB radio technology used at this stage was introduced in 2007 in the IEEE 802.15.4 standard for low-rate wireless networks as part of the IEEE 802.15.4a amendment. There, UWB is defined as an impulse-radio, meaning that transmissions consist of a sequence of very short (typically less than 2 ns) pulses. These short pulses, combined with the high bandwidth, enable excellent spatial resolution. This property enables precise estimation of a signal's transmission and reception timestamps, typically with picosecond resolution, making UWB an excellent candidate for time-of-flight (ToF)- based distance estimation.

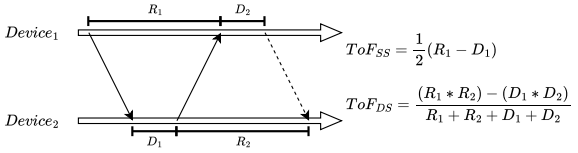


Figure 1: Ranging sequence of single- and double-sided TWR to determine the distance via the ToF by measuring the round-trip (R) and delay-times (D).

UWB frame structure. A UWB frame consists of multiple different sub-parts: First, there is a synchronization header (SHR) that itself consists of a *preamble* and a start-of-frame delimiter (SFD). The preamble is a ternary sequence of predefined pulses (i.e., positive, negative, or absent pulses) that are repeatedly transmitted to allow the receiving device to synchronise with the transmitter. The length of the preamble, i.e., the number of repetitions of the preamble symbol, is called preamble symbol repetitions (PSR). The SFD then marks the end of the SHR and the beginning of the *data-portion* of the frame. The second part, the data-portion, consists of a physical header (PHR) – containing general information such as the frame length and the data-rate of the remaining frame – and the actual payload with up to 127 byte of data. In contrast to the SHR where data is encoded in individual pulses, the information in the data-portion is encoded using a burst position modulation binary-phase shift keying (BPM-BPSK) scheme.

UWB-based ranging. The excellent time resolution of UWB enables a range of ranging and localisation techniques based on a signal's ToF that can be multiplied with the speed-of-light to get a distance. The presumably most widespread technique to obtain the ToF of a signal is two-way ranging (TWR). Due to its simplicity and the absence of tight synchronisation requirements, TWR is applicable in many applications. For TWR, at least two messages are exchanged between an initiator and responder (single-sided two-way ranging (SS-TWR)). By measuring the intervals between transmission and reception of the two frames, the distance between the two devices is estimated (see Figure 1). However, this technique is susceptible to clock drift, which is common in low-cost, low-power devices [26]. To prevent ranging errors due to clock-frequency offset (CFO), double-sided two-way ranging (DS-TWR) schemes can be used. By transmitting an additional message, the impact of response delays can be mitigated, and a more precise distance estimate can be obtained [26].

A primer on synchronous flooding. *Synchronous flooding* is a communication paradigm for wireless networks that has proven

its value, especially for low-power networks [34]. It is based on the concept of time-synchronised transmissions, in which nodes, after receiving a message, repeat and broadcast it simultaneously in a tightly coordinated fashion [42]. Concurrent transmissions of the same message lead to constructive interference and leverage the capture effect to enable highly reliable and efficient message propagation over multiple hops [1]. Starting with Glossy [11], synchronous flooding quickly became popular and has been utilised in various flooding primitives, such as Chaos [23], Crystal [18, 19], and Weaver [37].

OSF is a framework for synchronous flooding that enables the design of diverse flooding protocols [1]. Building on the Contiki-NG [14] operating system, it supports different narrowband radios from the Nordic nRF family. OSF supports multiple narrowband physical layers (PHYs) – i.e., narrowband IEEE 802.15.4 and different Bluetooth Low-Energy (BLE) PHYs such as 1M, 2M or coded PHY – allowing the protocol to be adapted to specific energy, reliability, and latency constraints. A protocol is defined as a sequence of *rounds*. An S-round is used to synchronise the participating nodes, while transmission (T) and acknowledgement (A) rounds are used to reliably exchange data. To improve robustness and gain multi-hop capabilities, each flood is repeated several times within each round. Using the so-called `ntx` parameter, one can configure the number of repetitions.

4 FLocк: Design and Implementation

In this section, we describe the design of FLocк and its different building blocks. We introduce the high-level design rationales of FLocк in Section 4.1 and give an overview of the system in Section 4.2. In Section 4.3, we introduce FLocк's ranging protocol and evaluate its scalability and the real-world performance in a testbed with up to 29 nodes. Section 4.4 describes how we use flooding to disseminate the obtained ranging information across the network and its scaling behaviour. Finally, in Section 4.5 we introduce the on-device localisation algorithm based on a gradient descent (GD) approach and evaluate its performance in simulated scenarios.

4.1 Design Rationale

We summarise below the fundamental design principles that guide the conception and implementation of FLocк.

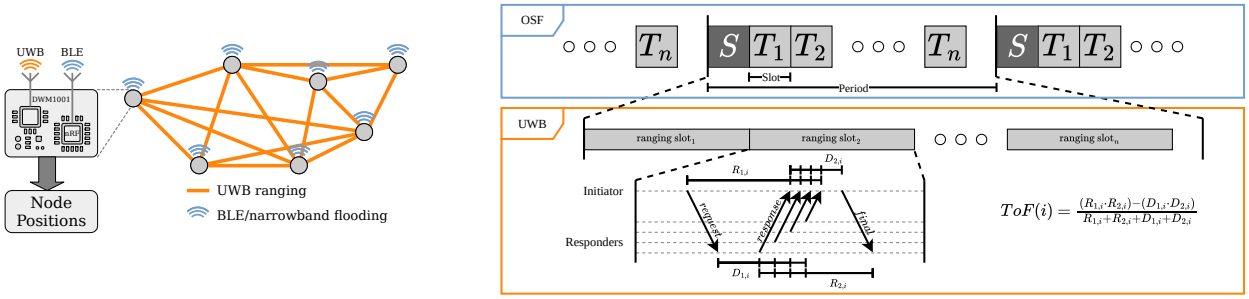
Precise, robust and efficient ranging. In a localisation system, performance is mostly determined by the accuracy and precision of the underlying ranging technology. Moreover, low-latency distance measurements are essential for supporting multiple mobile devices. Thanks to its excellent temporal resolution and its capability to *accurately* and *securely* determine distance information with *low-latency* using ToF, UWB is a natural choice for RF-based ranging. Several ranging paradigms have been proposed for UWB, including TWR and time-difference of arrival (TDOA) schemes. In FLocк, we adopt a TWR scheme for its simplicity and flexibility. Unlike TDOA, TWR schemes do not require tight nanosecond-level time synchronisation. Instead, nodes determine distances to neighbouring nodes using a straightforward handshake protocol (see Section 3). To ensure precise and robust distance estimates, FLocк employs a staggered variant of the DS-TWR scheme, which combines efficient ranging with high precision (see Section 4.3).

Dissemination of ranging information. Our selection of the communication backbone is guided by four key requirements: First, the backbone needs to be *multi-hop*-capable to cover large-scale areas and deployments. Second, it must be agnostic to *mobility* to support roaming nodes within the network. Third, the backbone should provide network-wide *synchronisation* to coordinate ranging slots across the network. Lastly, the backbone should be *self-sustaining* and not require additional infrastructure to operate independently. Although efficient and robust routing protocols have been widely studied in static sensor networks, these approaches often overlook mobility. We rule out infrastructure-based alternatives (i.e., WiFi or cellular technologies) because they depend on external infrastructure. Instead, FLocк leverages the broadcast capabilities of synchronous flooding schemes. Synchronous flooding has proven to be an efficient and reliable communication paradigm for low-power and lossy networks (LLNs), being inherently stateless – i.e., it does not rely on pre-established communication routes. In highly dynamic networks, the stateless nature of flooding maintains reliable communication without requiring explicit routing or network maintenance. Additionally, synchronous flooding provides network-wide synchronisation, which can be exploited to schedule UWB ranging slots. UWB-based flooding would be a promising candidate, due to its high data-rates of up to 27.2 Mbit/s, and has already proven its suitability for flooding [25, 37]. However, with FLocк, we relied on narrowband flooding to separate ranging from communication. This separation allows ranging measurements while simultaneously disseminating data across the network.

Real-time, on-device self-localisation. Although many algorithms have been proposed for self-localisation in radio networks, most are limited to single-hop deployments or require computationally intensive processing performed offline rather than directly on the resource-constrained device (see Section 2). While offloading computation enables the use of nearly unconstrained computational power, it also adds overhead, latency, and communication complexity. FLocк therefore explores a lightweight, multi-hop-capable, on-device solution. To this end, we employ a gradient descent (GD) approach that enables real-time processing of collected ranging information, even on constrained devices (see Section 4.5).

4.2 FLocк System Overview

Next, we provide an overview of FLocк's design before delving into the implementation details in the following sections. Figure 2a gives an overview of the system. With FLocк, several nodes form a localisation network. Thus, they make use of two radios. First, a narrowband (i.e., BLE) radio is used to synchronise the network and establish communication among all nodes via a synchronous flooding protocol built on top of OSF [1]. Section 4.4 provides details on how FLocк leverages flooding to disseminate the gathered distance information throughout the network. Second, an UWB radio is used to obtain distance information from all the neighbours in the vicinity. Section 4.3 will detail how a staggered ranging approach can be utilised to efficiently gather this information. Sharing distance information among all participants in the network via the OSF backbone allows each node to compute a position estimate for the entire network using the on-device localisation engine (see Section 4.5).



(a) Overview of a FLocк network. **(b) FLocк's protocol for the data dissemination and ranging.**

Figure 2: An overview of FLocк. An OSF-based flooding protocol is used for synchronisation and data dissemination. Aligned ranging slots are used by the nodes to initiate rangings to their neighbours. Using the measured and shared distance estimates, each node can compute the positions of all nodes within the network.

Hardware and software implementation. We implement FLocк on the DWM1001-DEV board from Qorvo. These developer boards feature an nRF52832 MCU and a DW1000 UWB transceiver, attached via SPI. The Bluetooth 5.4-compliant radio in the MCU is used for narrowband flooding, whereas the UWB transceiver provides range measurements.

FLocк builds on top of the OSF framework, which itself is built on the Contiki-NG [14] operating system. For FLocк, we added a DW1000 driver to Contiki-NG and implemented a DS-TWR ranging scheme (see Section 4.3), the OSF-based data-dissemination protocol (see Section 4.4), and the on-device localisation algorithm (see Section 4.5). The on-device localisation has been implemented as a Contiki-NG process, whereas time-critical tasks such as OSF and UWB radio tasks are interrupt-based and can therefore always be executed preemptively.

4.3 Staggered Two-way Ranging

In the following, we describe the used ranging scheme and evaluate its performance in terms of accuracy and precision, as well as its scaling behaviour.

Ranging scheme. In FLocк each node has a contention-free *ranging slot* in which it can initiate rangings to its neighbours. The slot alignment is derived from the synchronisation provided by OSF, and the slot assignment is determined by the node's OSF ID. To determine the distances among the nodes, we use a staggered variant of the DS-TWR scheme. Figure 2b contains the ranging protocol. Each node has its own ranging slot, during which it initiates a ranging round by broadcasting a *request* message containing the desired ranging partners, randomly chosen from its currently known neighbours. The request message serves a dual purpose as a beacon message for neighbourhood discovery, so that all listening neighbours can update their neighbourhood tables accordingly. Hence, also with no known neighbours, an empty *request* must be broadcast. The ranging partners sequentially respond – in the order indicated by the *request* – with a *response* message that contains no additional payload. After receiving the RESPONSE messages, the initiator broadcasts a *final* message containing the ranging data for the neighbours to complete the ranging. As the DW1000 is known to be susceptible to range bias due to varying received signal strength, we apply the vendor-provided range-bias correction.

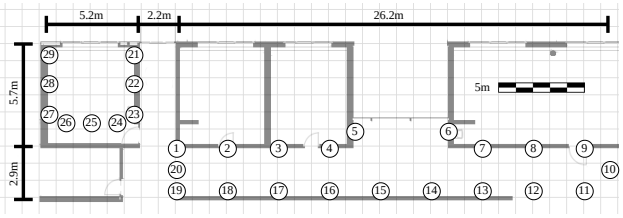


Figure 3: Sketch of the testbed used for the experiments.

Support for different UWB-PHY configs. UWB offers a variety of PHY settings to tune the performance to specific needs [15]. Some of these parameters, i.e., the data rate or the length of the PSR, have a significant impact on the transmission time of a frame. Hence, the timing of FLocк's ranging scheme can be adapted to different configurations to optimise the system for update rate, robustness, or energy efficiency. In this paper, we investigate two different PHY configurations: one with a short preamble of length 64 (PSR64) and one with a long preamble of length 1024 (PSR1024). While PSR64 minimises the air time (max. 250 µs vs. 1.2 ms) and thereby allows higher update-rates, PSR1024 ought to improve the robustness of the preamble and therefore the distance estimate. If not explicitly stated otherwise, we use PSR64 as our default configuration. Other than that, both configurations use channel 5 with a pulse-repetition frequency (PRF) of 64 MHz and a data rate of 6.8 Mbit/s.

Ranging performance. Next, we evaluate the accuracy and precision of the ranging scheme in a real-world deployment using a testbed facility at our premises (see Figure 3) [32]. The 29 nodes are distributed across an OFFICE and a CORRIDOR. Ground-truth positions for the nodes were obtained using a Leica Nova MS60 total station theodolite. We deploy FLocк and let the nodes perform range measurements among themselves using both PHY configurations. More than 200 measurements have been taken for each node pair. The matrix in Figure 4 shows the median absolute error of all ranging pairs. While there are no links between the office and the hallway environment, most node pairs have offsets well below 30 cm. The outliers with errors above 30 cm mainly stem

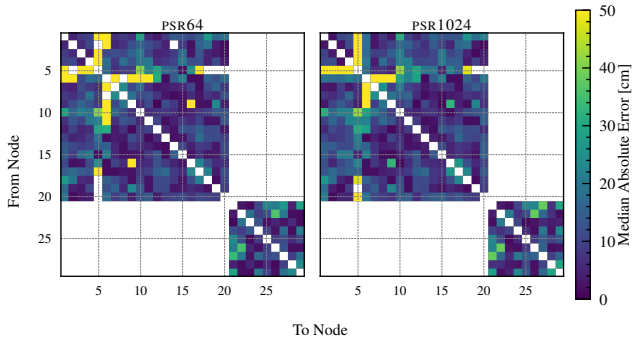


Figure 4: Median absolute error of the distance estimates for the two different PHY configurations.

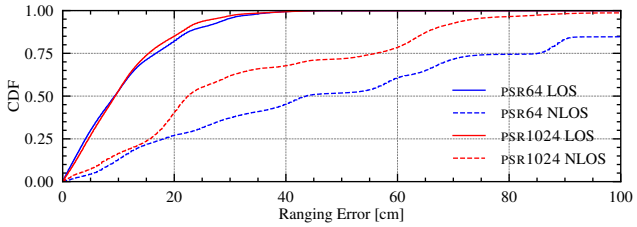


Figure 5: CDF of the observed ranging error in the testbed for both PHY configurations and manually labelled for LOS/NLOS.

from NLOS conditions concerning distance measurements with nodes (5) and (6). An interesting observation is the node pair ((9), (16)), despite clear line-of-sight (LOS), suffers from outliers in both directions and for both PHYs (PSR64 and PSR1024) with 58 cm and 33 cm median absolute error, respectively. We attribute this to an unfortunate placement and multipath components that might lead to destructive interference [15].

Figure 5 shows the cumulative distribution function (CDF) of the ranging error. We manually label all node pairs in the testbed, distinguishing between LOS (86.90 % of the pairs) and NLOS. Under LOS the ranging performance of both PSR64 and PSR1024 is similar, with a median absolute error of 9.3 cm and 9.4 cm respectively, and a 99th percentile of 38.1 cm for both configurations. In the case of NLOS however, we see a significant impact of the longer preamble, with a median absolute error of 22.5 cm (PSR1024) versus 43.8 cm (PSR64) and a 99th percentile of 3.68 m and 11.34 m respectively. Focusing on the LOS rangings, the distribution of the ranging errors follows a normal distribution with $\sigma = 13.7$ cm and $\mu = -3.8$ cm.

Ranging update rate and scalability. The scalability of the ranging scheme is limited by the payload length of the UWB frame. A standard frame in the IEEE 802.15.4 can carry up to 125 bytes of payload (127 - 2 bytes for a checksum). Losing five bytes to message headers – which includes the node addresses and a frame type – leaves us with 120 bytes of payload. The critical part of the ranging is the *final* message, which contains the measured R_1 and D_2 intervals. Each of these timestamps needs 4 bytes, hence limiting the number of ranging partners per slot to 15. Using the non-standard extended-length data frame with up to 1023 bytes offered by the DW1000, one can increase the number of ranging partners per slot to 127.

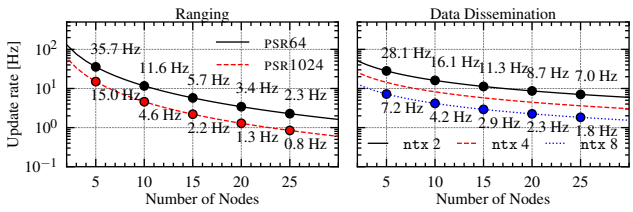


Figure 6: Achievable update rate for the data-dissemination and 'all-to-all' rangings calculated based on the configuration's timing constraints.

Regarding the update rate of the ranging scheme, we must account for the varying PHYs. We define an update as a full set of 'all-to-all' measurements within the network. For example, in a ten-node network, this means measuring 90 distances. Figure 6 shows the theoretical update rate for the two different PHY configurations based on their timing requirements. Using PSR64 with 10 nodes results in an update rate of 11.6 Hz, whereas with PSR1024 only 4.6 Hz could be achieved. Notably, all-to-all rangings are the upper bound on the ranging information that can be gathered within a network and, hence, represent a worst-case scenario with respect to the update rate. Especially in multi-hop scenarios with limited connectivity, foreseeing all-to-all measurements is excessive.

4.4 Flooding-based Data Dissemination

To this end, data dissemination and collection using OSF have been utilised only for one-to-all and all-to-one connections [1]. These tasks are often implemented via an 'STA' protocol: after a sync-round (S-round), a single transmission (T-round) is performed, which is acknowledged by the receiver afterwards (A-round). This pattern has proven to be very effective and robust for sensor networks, especially when data is generated aperiodically. With this protocol, collisions in the T-round can be easily detected and resolved in the A-round and mitigated via a random backoff [1].

With FLocк however, we expect new ranging data from each node in each period. Hence, using a STA protocol would lead to numerous flood collisions and limit the system's throughput. A group acknowledgement, i.e., a mechanism that verifies the successful reception on all participating devices, is not feasible for a dynamic network where participants could leave the network at any time. Instead, we leverage a novel STT protocol (see Figure 2b) in which, after a sync round (S), each participating node has its own contention-free timeslot (T) to broadcast its latest ranging data. Consequently, we can rely on the redundancy of the flooding scheme to ensure data is distributed to all participants. Figure 6 illustrates the relation of the achievable update rate for an increasing number of nodes and an increasing number of flood retransmissions (`ntx`). In OSF, by default, per `ntx`, the maximum number of hops is doubled, leading to a up to four-hop network for `ntx` = 2 and a 16-hop network for `ntx` = 8. We thereby define the update rate as the minimum period length of the STT protocol, with every node having a contention-free slot. To minimise latency in data dissemination, we use the 2M PHY with an `ntx` of 2, which is sufficient for our deployments.

4.5 On-device Localisation Engine

Localisation is performed via an iterative network adjustment. The goal is to find node positions that minimise the sum of the squared residuals. Hence, we define the objective function for our optimisation problem as

$$\min_{P} \sum_{d_{ij} \in D} (||p_i - p_j|| - d_{ij})^2$$

with $d_{i,j} \in D$ being the measured distance between node i and j and $p_i \in P$ being the estimated position of a node i.

With the GD method, we iteratively approach this minimum. For each step, we iterate over all observed distances, compute the gradient of the node positions, and update them. The GD method has two hyperparameters. The maximum number of iterations T performed and the learning rate α. While T specifies how many steps will be taken for the algorithm, α determines the size of the step along the calculated gradient.

Depending on the quality of the initial guesses and the coherence of the distance measurements, the algorithm might converge faster or slower. In each iteration, we track convergence by the mean residual R and its improvement. If the improvement in an iteration falls below a defined threshold δ for multiple consecutive iterations, we assume the algorithm has converged and terminate before reaching the maximum number of iterations.

Estimation quality and resolving ambiguities. Furthermore, the final value of R yields information about the quality of the estimate. If the estimated points fit the observations, R converges to the expected measurement deviation. In contrast, if the observed deviation differs significantly from the expected one, it is likely that the algorithm converged to an ambiguous solution or that the measurements contain faulty distances e.g., due to NLOS. We use this information to rerun the GD algorithm with a new set of initial positions.

Chosen default parameters. If not explicitly stated otherwise, we set the learning rate $\alpha = 0.02$ and the number of maximum iterations $T = 500$. Further, we set the early break threshold $\delta = 1$ mm and expect the average residuals R to be below 25 cm, which is roughly twice the σ of our ranging measurements (see Section 4.3). If R is larger than the expected value, we rerun the GD with a different set of initial positions up to 16 times. If, after 16 retries, no satisfactory solution could be found, we consider the localisation to have failed and return the position estimate from the best run.

Simulation and comparison to other algorithms. Next, we test the on-device engine using simulated distance information across different scenarios to assess its performance in diverse setups. Using a Python script, we generate various random scenarios, simulate the distance measurements, and send them via serial to a test node running the on-device engine. For the tested scenarios, we randomly place an increasing number of nodes (15, 30, 45) in square environments (10 m, 20 m and 30 m edge length). We ensure that the resulting network is connected and select four known positions at the network's edge to reduce the possibility of ambiguities [5]. The known positions are selected automatically by calculating the network's centre of mass, dividing it into quadrants, and selecting, in each quadrant, the node with the largest distance to the centre. Furthermore, we restrict the distance measurements to 15 m, resulting in single-hop scenarios in the 10 m × 10 m case and multi-hop

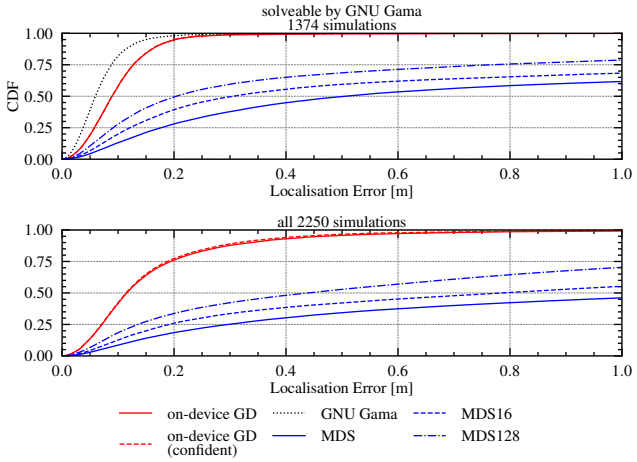


Figure 7: CDF of the localisation error in the simulations. While the first plot only includes simulations that were also solved by GNU Gama, the second shows the distribution of all 2250 simulations. Results for the on-device GD with confidence refer to solutions that meet our quality criterion (average residual ≤ 25 cm) within 16 retries.

scenarios with up to three hops in the other environments. To mimic our real-world measurements, we add normally-distributed noise to the distance measurements with $\sigma = 13.7$ cm and $\mu = -3.8$ cm according to our ranging measurements (see Section 4.3). For each environment, we test 25 random arrangements and repeat each one 10 times with new distance estimates, resulting in a total of 2250 simulations. For fairness, we clear the connected device's memory between each simulation to ensure that previous estimates do not affect subsequent runs.

We compare the on-device engine with two different off-device solutions: first, GNU Gama [39], an open-source package intended for geodetic surveying and second, the publicly available MDS implementation provided by Koledoye et al. [21] that extends the MDS implementation of the Python `scikit-learn` package similar to the proposed solution of Corbalán et al. [5]. Following the suggestions from Corbalán et al. [5], we run the MDS multiple (1, 16, 128) times with different initialisation values to reduce the impact of ambiguities. While MDS and also our on-device approach always yield a result, due to its background in geodetic applications, GNU Gama does not return a solution e.g., if there are unresolved ambiguities in the network configuration or if a network is underdetermined. Hence, based on the results, we have to distinguish between two cases: first, we consider all simulated scenarios (2250 simulations); second, only scenarios in which GNU Gama also returned a result (1374 simulations). Furthermore, we differentiate for our on-device localisation if the engine has been confident in the result (i.e., if the mean residual R is ≤ 25 cm). Figure 7 shows the CDFs of the localisation errors, obtained by the different algorithms. Focusing on the scenarios with a successful GNU Gama estimate, we can see that GNU Gama outperforms the other algorithms with a mean absolute error (MAE) of just 6.9 cm, compared to the 10.3 cm of the on-device GD (10.1 cm with confidence) and the 75.9 cm of the best performing MDS with 128 runs (MDS128). The 95^{th} percentiles vary from 15 cm (GNU Gama) to 20.2 cm (on-device GD) and 20 cm

(on-device GD with confidence). In our simulations, the MDS variants are not robust to ambiguities and poor network configurations, leading to a 95^{th} percentile from 3.398 m (MDS128, 128 repetitions) to more than 13.8 m (MDS with single run).
Considering all tested scenarios, including those that could not be solved by GNU Gama, we see that overall performance deteriorates. Over all 2250 simulations, the on-device GD achieved a MAE of 17.2 cm (15.8 cm with confidence), compared to 94.6 cm of the best performing MDS (MDS128). Also the 95^{th} percentile increases from 46.4 cm and 43 cm (with confidence) of the on-device GD to 3.461 m for MDS128.

Runtime complexity and scaling behaviour. Next, we examine the runtime behaviour of the on-device GD algorithm to assess the feasibility of real-time on-device self-localisation. The runtime of the algorithm depends on two main factors: First, the number of distance observations $|D|$. Let n be the number of nodes in our network; the maximum number of edges is $n * (n-1)$. As our distance measurements are directed, for each edge, we obtain up to two observations ($d_{ij} \neq d_{ji}$), leading to a maximum of $2n*(n-1) \in O(n^2)$ observations to be processed. Second, the maximum number of iterations T is a constant. While an early stopping criterion can be used to terminate the algorithm earlier, a maximum bound T ensures termination. As T is a constant, the complexity of the localisation lies within $O(n^2)$.
For real networks, runtime will be strongly influenced by the layout and by the quality and consistency of the measurements. To assess the runtime of the on-device GD algorithm, we measure its runtime during the simulation and compare it with the runtime of the off-device GnuGama and MDS implementations. Notably, while the MDS implementation is purely Python-based and thus likely to experience some performance degradation, GNU Gama is written in C++ and can therefore directly leverage the CPU's capabilities. Both off-device algorithms are executed sequentially on an Intel i7-8565U CPU. Instead, the on-device algorithm runs on the Nordic nRF52832 MCU of the DWM1001-dev board. For on-device GD, we measure runtime using the MCU's timer with a resolution of 16 µs. During the experiment, the node was not connected to a FLocк network, so interrupts from OSF or the UWB radio could not affect the runtime. For the on-device GD, we timestamp just before initialisation and just after the last retry's termination. For the off-device algorithms, we track calculation time using system time, excluding additional overhead such as data formatting as much as possible.
Figure 8 shows the scaling behaviour of the algorithms for an increasing number of nodes. The median runtime of the GD increases from 418.7 ms (15 nodes), to 3.222 s (45 nodes). If we prime the GD with the estimates from the previous runs, this runtime can be significantly reduced to a range between 42.6 ms (15 nodes) and 148.8 ms (45 nodes). The reason for the long runtime of the clean GD can be seen in two metrics: First, the number of retries, with a mean of 0.83 for clean runs compared to a mean of 0.07 for primed ones; second, the average number of iterations of the GD increases from 41.3 for the primed runs to 228.6 for the cleaned runs.
Focusing on the comparison values from GNU Gama, it is evident that a pure C++ implementation on a laptop yields a runtime advantage compared to the on-device GD method running on the MCU.

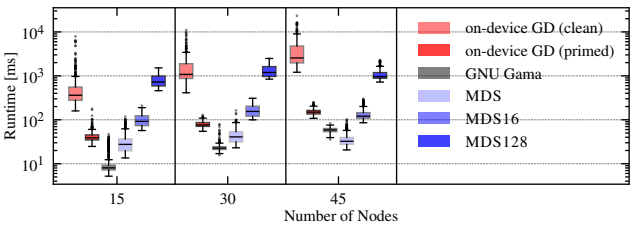


Figure 8: Runtime comparison of the different algorithms during the simulation for an increasing number of nodes. The on-device GD has been executed in two variants with different initialisation coordinates: *'clean'*, with random coordinates for each run, and *'primed'*, with initialisation coordinates taken from the previous run's results.

The runtime rises from only 8.04 ms (15 nodes) to just 58.71 ms. However, it can also be seen that GNU Gama's runtime increases faster than that of the on-device GD method as GNU Gama solves the system of normal equations whose complexity scales with the cube of the number of nodes ($O(n^3)$).
Interestingly, for the MDS variants, we observe a less pronounced increase in runtime with increasing node count. For the single-run MDS, the runtime increases from 29.09 ms (15 nodes) to 40.81 ms (45 nodes) and from 808.20 ms to 1237.51 ms for MDS128, respectively. We conjecture that this flat rise stems from the increasing number of ambiguities as the number of nodes increases in a multi-hop scenario. Running into ambiguities could trick the MDS into believing that a solution has been found, thereby terminating the optimisation. This observation also aligns with the localisation performance, showing large outliers for MDS in complex scenarios due to ambiguities.

5 FLocк in the Real World

In this section, we evaluate the performance of FLocк in real-world conditions. We first investigate the performance of FLocк in two static environments, office and corridor (see Section 5.1). Next, shifting to mobile setups, we demonstrate FLocк's ability to track nodes across multiple hops, and present an example of relative localisation compared with the ground truth from a motion capture (MOCAP) system (see Section 5.2).

5.1 FLocк in Static Environmnents

Next, we evaluate the performance of FLocк in the same testbed environment as the range measurements (see Section 4.3). As shown in the ranging measurements (see Figure 4), the testbed is split into two disjoint environments using UWB: an office and a corridor. In the following, we will evaluate these environments separately using both PHYs (psr64 and psr1024) and enforce multi-hop connectivity within the corridor by filtering out UWB ranges above 10m. For the experiment, we run FLocк on the devices with an OSF period length of 1 s, resulting in an update rate of 1 Hz. For each configuration and setup, we make 3000 localisation attempts. In addition to the position estimates, we track the reported average residual of the gradient-descent algorithm (R) and the number of retries and iterations required. A summary of the experiments is provided in Table 2. In the following, we discuss the individual setups in detail.

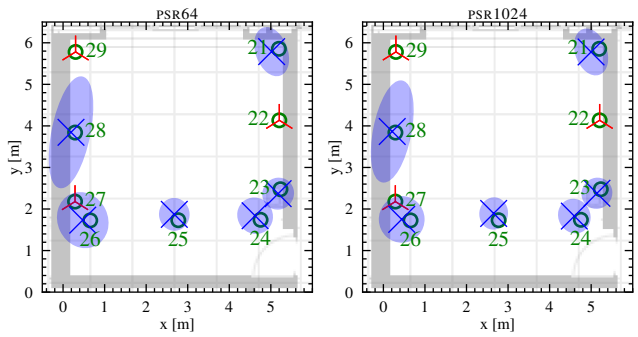


Figure 9: Localisation in the OFFICE for the two different PHYs configurations. Green ○ indicates the ground-truth position of the nodes. Blue × mark the mean estimates surrounded by blue ellipses indicating the 95^{th} percentile of the χ^2 distribution scaled by a factor of 20. Red ⅄ mark the known positions.

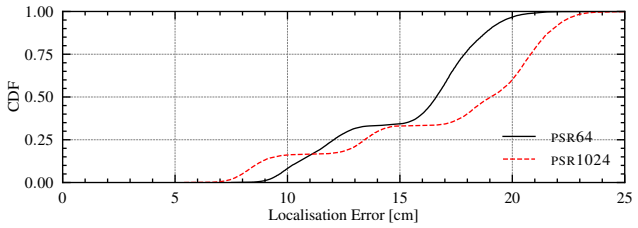


Figure 10: CDF of the localisation error in the OFFICE for the two different PHYs configurations.

FLOCK in the OFFICE. First, we run FLOCK in the office with nine nodes (nodes (21)-(29) in Figure 3) of which three positions are known ((29), (27) and (22)). This setup is considered simple, as all nodes are deployed along the walls of a nearly square room in a single-hop environment, with clear LOS. The maps in Figure 9 show the mean estimated node positions, along with error ellipses that depict the 95^{th} percentile of the estimates, magnified by a factor of 20 for visibility. Figure 10 shows CDF of the according localisation error. As can be seen, in this favourable environment, the localisation performance of the two different PHYs matches within a couple of centimetres. While PSR64 yields an MAE of 15.4 cm, PSR1024 obtains 17.1 cm MAE with a 95^{th} percentile of 19.7 cm and 22.4 cm respectively. The similarity in performance is also evident in the average residual reported by the localisation algorithm, with 12.3 cm for PSR64 and PSR1024, respectively. For both PHY, the algorithm immediately found a solution without any retries.

FLOCK in the CORRIDOR. Next, we take a look at the performance in the 20-node setup in the CORRIDOR (nodes (1)-(20)). We select four known positions ((1), (9), (11) and (19)) at the edge of the network to reduce the possibility of ambiguities [5]. Here, the nodes are mostly arranged symmetrically opposite one another along a corridor, creating a more challenging environment. We conjecture that measurements among nodes placed mostly along a line yield valuable information in one dimension but not in the other, leading to a high geometric dilution of precision (GDOP). Hence, we expect a higher resolution along the corridor than across its width.

The first column in Figure 11 shows the localisation performance for PSR64 and PSR1024 while the CDF of the localisation error can be seen in Figure 12 (solid lines). In this environment, the configuration with the short preamble leads to severe outliers. PSR64 yields a MAE of 2.818 m and 95^{th} percentile of 9.280 m, while the PSR1024 configuration achieves a MAE of 33.2 cm with a 95^{th} percentile of 65.9 cm due to its more robust distance estimations (see Section 4.3). The decreased localisation performance is also evident in the estimated average residuals: while PSR1024 converges to an average of 27.4 cm, PSR64 achieves only 1.884 m.

Impact of NLOS. To verify that the inferior performance of PSR64 is indeed due to NLOS, we next remove the affected nodes ((5) and (6)) from the setup. The second column in Figure 11 shows the localisation performance of the remaining nodes with the corresponding CDFs in Figure 12 (dashed lines). Indeed, for PSR64 we can see an improvement of the localisation error to 26.0 cm (from 2.818 m with NLOS) while for PSR1024 the MAE reduces to 22.2 cm (from 33.2 cm). For the 95^{th} percentile this leads to a reduction to 78.01 cm (PSR64) and 59.15 cm (PSR1024).

From single- to multi-hop. To assess the performance in a multi-hop setup, we next limit the range of the UWB distance estimates by filtering distance measurements above 10 m. Therefore, the CORRIDOR environment extends to a multi-hop setup with up to three hops. By removing (5) and (6), we focus on the LOS measurements, as distance-based filtering would affect the NLOS performance of our measurements. The third column in Figure 11 shows the localisation performance for this setup, with the CDF of the localisation error being depicted by the dotted lines in Figure 12. As shown in the CDF, the localisation performance decreases slightly compared to the single-hop scenario, with a MAE of 23.2 cm (PSR64) compared to 36.0 cm (PSR1024) and a 95^{th} percentile of 43.6 cm and 95.2 cm. Notably, the outliers in the PSR1024 case have been caused by 'flipping' nodes (e.g., of nodes (7) and (13)) caused by the network's symmetric geometry and the GDOP.

Table 2: Summary of the localisation performance.

	OFFICE		CORRIDOR single-hop		CORRIDOR single-hop LOS only		CORRIDOR multi-hop LOS only	
	PSR64	PSR1024	PSR64	PSR1024	PSR64	PSR1024	PSR64	PSR1024
RMSE [cm]	15.76	17.78	401.26	49.08	45.30	36.46	38.26	85.14
MAE [cm]	15.41	17.11	281.80	33.19	25.98	22.24	23.17	35.96
95^{th} [cm]	19.66	22.36	928.04	65.88	78.01	59.15	43.67	101.02
R [cm]	12.29	12.29	188.39	27.45	16.55	22.07	9.34	19.40
#iterations	9.53	9.18	227.14	21.85	16.44	17.03	16.24	29.32
#retries	0.00	0.00	15.78	1.27	0.08	0.72	0.02	0.59
success [%]	100	100	1.40	92.03	99.53	95.53	99.90	96.37

5.2 From Static to Mobile Setups

We next investigate FLOCK in two mobile scenarios. First, we show FLOCK with a roaming node traversing multiple hops within the network. Second, we show how FLOCK can be used for relative localisation with five nodes.

Multi-hop roaming within FLOCK. To demonstrate FLOCK's ability to seamlessly track nodes across multiple hops, we deploy FLOCK on the testbed with the two disjunct UWB domains OFFICE and CORRIDOR (see Figure 4). To simulate multiple hops, we reuse the CORRIDOR's multi-hop setup from Section 5.1. Notably, even though OFFICE and CORRIDOR are disjunct UWB domains, they still form a

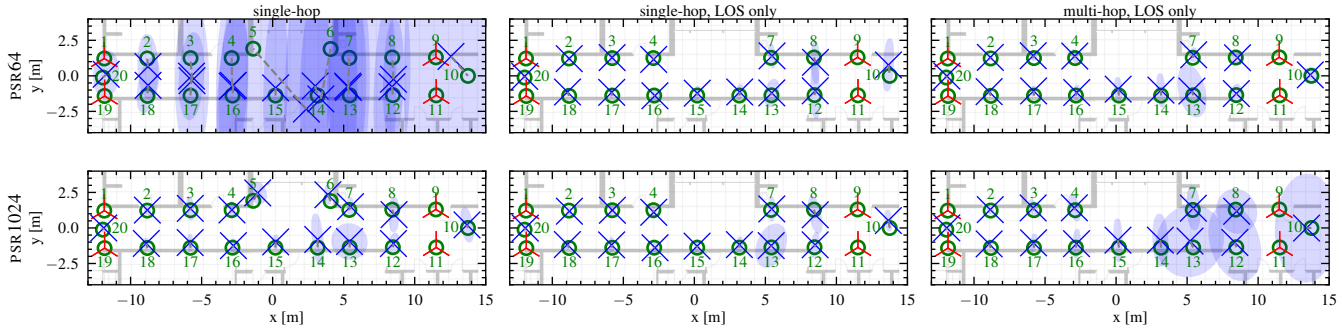


Figure 11: Localisation performance in the CORRIDOR. Green ○ displays the ground-truth positions. Blue × mark the mean estimates. Blue ellipses indicate the 95^{th} percentile of the χ^2 distribution. Red 人 mark known positions.

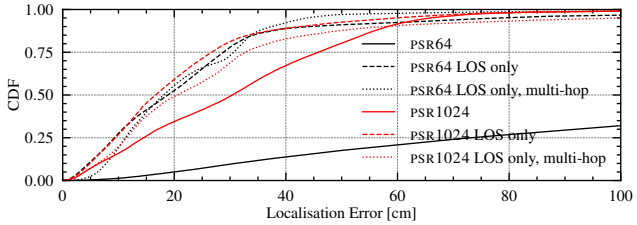


Figure 12: CDF of the localisation error in the CORRIDOR for the three tested scenarios and two configurations.

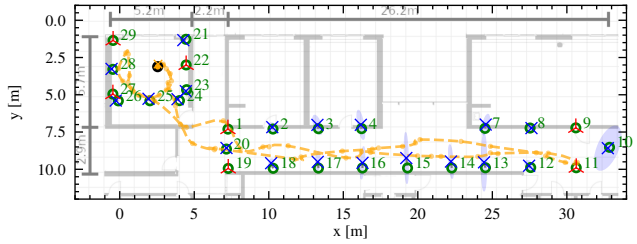


Figure 13: Localisation of a roaming device in a multi-hop configuration. The orange line indicates the estimated path of the mobile node.

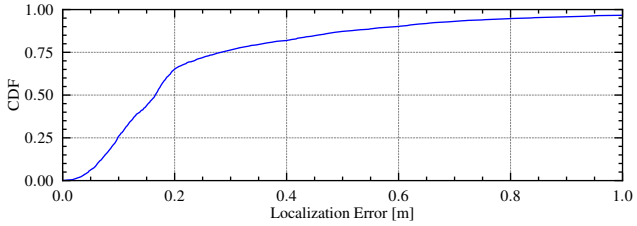


Figure 14: CDF of the static nodes in the roaming experiment.

FLOCK and share all distance measurements via the OSF backbone. During the experiment, we move a mobile node carried by a person at walking speed through the OFFICE in a half-circle, then up and down the CORRIDOR. On its path, the mobile node traverses across four different hops (one hop OFFICE, three hops CORRIDOR) and even seamlessly roams between two disjunct UWB domains.

Figure 13 shows the setup and the calculated path of the mobile node estimated by FLOCK. We track errors on the static nodes in the testbed to assess the impact of moving a node. The CDF in Figure 14 shows the distribution of the error of the static nodes during the experiment. Compared to the static evaluation in the CORRIDOR, we

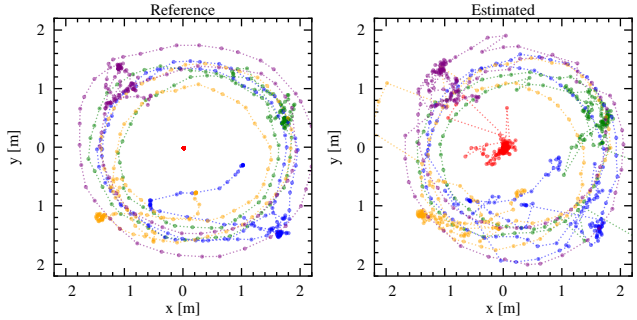


Figure 15: Relative localisation in a single-hop setup with five nodes. Plots show the reference positions of the nodes in different colours, and the corresponding estimates of FLOCK.

see a slight decrease in the localisation performance from a MAE to 26.8 cm from 23 cm and a increase in the outliers with a 95^{th} percentile increasing from 43.6 cm to 83.1 cm.

Relative localisation in highly mobile scenarios. Next, we investigate the relative localisation capabilities of FLOCK. For this experiment, we use a small-scale setup in which all node positions can also be tracked by a Qualisys MOCAP system with sub-millimetre accuracy at an update rate of 100 Hz. The test environment is a 4 m × 4 m environment that is tracked by 8 Miqus M3 cameras. The accuracy of the MOCAP system after calibration is reported to be 0.76 mm. For the experiment, we use five DWM1001-dev boards. One, placed in the centre of the area, is connected to a laptop via serial and reports the relative position estimates with an update rate of 3 Hz and hence remains stationary. The four remaining devices have been carried by people roughly at the same height as the stationary one and moved around it in two circles (clockwise and counterclockwise) within a minute. On the arrival of a new measurement, the laptop retrieves the ground-truth positions of the nodes from the MOCAP and compares them with the FLOCK's estimates. To compare the estimated positions with the ground truth, we use the Kabsch algorithm [20] to rotate each estimate and align it with the ground truth. Thus, we fit the relative estimate to the absolute ground truth without altering the network's layout.

Figure 15 shows the localisation performance of the relative localisation and the comparison to the ground-truth. The corresponding CDF of the experiment can be seen in Figure 16. As shown, FLOCK has reconstructed the network's movement with a MAE of 19.3 cm,

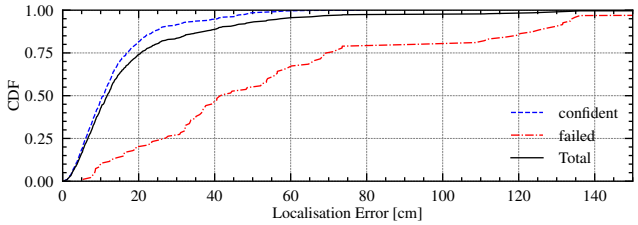


Figure 16: CDF of the relative localisation error of the on-device localisation separated between confident (average residual $\leq$ 25 cm) and failed localisation attempts.

a RMSE of 32.7 cm and a 95^{th} percentile of 57.4 cm. During the experiment, 79 % of the estimates are within the confidence criterion of the GD algorithm i.e., within the expected two *sigma* (25 cm) limit of the average residual R. Due to the dynamic setup, it is not possible to control the current geometry and, consequently, the network's stability. Furthermore, individuals carrying the device impact the measurements by creating NLOS conditions. Focusing on the failed attempts, with R exceeding 25 cm, reveals a MAE of 60.2 cm compared to 13.6 cm for the confident cases. However, since the GD method is capable of identifying these uncertainties, it proves to be robust in the face of erroneous measurements. During the experiment, on average, the GD took 51.7 ms with an average of 35.7 iterations and 3.38 retries.

6 Discussion and Future Work

FLock presents a step towards the self-localisation of multi-hop networks, opening future directions.

Leveraging next generation of UWB devices. While FLock has been implemented as a proof-of-concept on a well-established UWB platform due to availability reasons, the next generation of UWB radios offers new and interesting capabilities for network self-localisation. The introduction of the IEEE 802.15.4z-amendment improved the security of UWB range measurements through randomised pulse sequences (scrambled timestamp sequenece (STS)). While security has been out of scope of this work, future work will incorporate security to foster FLock against attackers.

Integration of additional location information. The integration of angle-of-arrival features in modern multi-antenna UWB-receivers not only enables accurate distance measurements among devices but also provides angular information. Recent work from Garg et al. [13] demonstrated how additional angular information can be used to reduce the total number of required range measurements. Hence, leveraging angular information within FLock's localisation engine could not only improve the quality of the network estimate but also resolve ambiguities in the network and reduce the number of required measurements.

In addition to UWB measurements, modern mobile devices such as smartphones and wearables offer a wide variety of motion sensors that can be used to refine the position estimate, reduce ambiguities and decrease localisation requirements such as the number of known positions. Additionally, incorporating a notion of time into FLock could improve the quality and robustness of localisation. Hence, future work will investigate how to integrate additional location information and the implications for the required computational and communication overhead.

Robustness against NLOS. Our real-world experiments showed FLock's susceptibility to erroneous measurements, for example, due to NLOS. Therefore, future work will investigate how to incorporate on-device classification and mitigation techniques, such as lightweight machine learning approaches [12]. Furthermore, introducing extensions such as weighting and outlier detection into the GD algorithm is a promising approach to enhance the robustness of FLock's estimates under challenging real-world conditions.

From 2D to 3D. In this paper, we focused only on localisation in two dimensions. While FLock's principles could easily be adapted to three dimensions, the additional degree of freedom increases ambiguity, complicating the establishment of a unique network.

FLock in diverse, highly mobile deployments. This paper quantifies FLock's performance across different environments and deployments and investigates it in various simulated scenarios. However, due to the lack of automated testing facilities for dynamic, large-scale deployments, our mobility experiments are limited in both size and the number of nodes. Hence, deploying FLock on a large-scale, dynamic, multi-hop network i.e., a drone swarm with dozens of flying nodes, would be an interesting challenge.

Towards a flock of FLocks. In this paper, we showed that FLock allows networks to scale from small-scale deployments with only a handful of devices towards larger-scale deployments with dozens of devices. However, in the current state, the scalability of FLock towards hundreds of devices is limited by two major obstacles. First, the communication backbone: As flooding creates a network-wide shared medium, contention-free slots for each device limit scalability with linear growth. Second, the quadratic growth of the ranging data increases computational costs, which is especially concerning for devices with limited computational power. Hence, in future work, we want to investigate how to overcome these limitations, i.e., by splitting the network into (sub-)clusters that could cooperate to form a flock of FLocks. Thus, each cluster could share and process its ranging data within the cluster, reporting only the calculated positions to the entire network. As a result, both communication and computational effort could be distributed across the different clusters. However, this approach will require new hierarchical flooding paradigms for communication and will pose interesting management challenges, such as device roaming among clusters, that warrant investigation in future work.

Need for benchmarking. Although there are already several solutions to the problem of infrastructure-free multi-hop localisation (see Section 2), there is a lack of comparisons among them. Unfortunately, existing work so far has merely been made publicly available, or has been designed with specific hardware requirements. Hence, future efforts should focus on comparing the various solutions head-to-head in repeatable and fair environments.

Considering energy efficiency. To this end, we have not prioritised energy efficiency in FLock's implementation. However, due to the high energy demand of UWB radios, energy efficiency can be key for energy-constrained applications. Improvements such as leveraging energy-efficient narrowband for neighbourhood discovery and scheduling measurements only when needed e.g., if movement has been detected, could save energy and thereby prolong node lifetimes. Furthermore, sharing estimated positions among

devices would allow devices with even less computational power or limited energy to obtain position information via the network.

7 Conclusion

In this paper, we introduced FLock, a localisation system leveraging the excellent UWB ranging capabilities alongside the robust and flexible communication properties of narrowband flooding. By broadcasting the measured distances among all nodes in the network, FLock efficiently collects the information required to enable self-localisation of the entire network. A lightweight on-device localisation engine is presented that enables the network to self-localise in real time. Combining these properties FLock presents a holistic approach for UWB-based network localisation that collects, disseminates, and processes ranging data within a single system. The source code for FLock is available as open source for future extensions [33].

Acknowledgments

The authors would like to thank Peter Bauer and the Institute of Geodesy for providing ground-truth measurements for the testbed nodes. We would also like to thank Wolf-Dieter Schuh for its advice and discussions regarding the localisation algorithm. This work has been performed within the SPiDR project ("Secure, Performant, Dependable, and Resilient Wireless Mesh Networks") financed by the Technology Innovation Institute.

References

[1] Baddeley, Michael et al.. 2022. OSF: An Open-Source Framework for Synchronous Flooding over Multiple Physical Layers. In *Proceedings of the 19^{th} Conference on Embedded Wireless Systems and Networks.*

[2] Bulusu, Nirupama et al.. 2002. GPS-less low-cost outdoor localization for very small devices. *IEEE personal communications* (2002).

[3] Cao, Yanjun et al.. 2018. Dynamic range-only localization for multi-robot systems. *IEEE Access* (2018).

[4] Cao, Yanjun et al.. 2021. Distributed TDMA for mobile UWB network localization. *IEEE Internet of Things Journal* (2021).

[5] Corbalán, Pablo et al.. 2023. Self-localization of ultra-wideband anchors: From theory to practice. *IEEE Access* (2023).

[6] Dhekne, Ashutosh et al.. 2019. TrackIO: Tracking first responders Inside-Out. In *Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation.*

[7] Di Franco, Carmelo et al.. 2017. Calibration-free network localization using non-line-of-sight ultra-wideband measurements. In *Proceedings of the 16th Conference on Information Processing in Sensor Networks.*

[8] Di Franco, Carmelo et al.. 2017. Multidimensional scaling localization with anchors. In *Proceedings of the International Conference on Autonomous Robot Systems and Competitions.*

[9] Di Franco, Carmelo et al.. 2018. Dynamic multidimensional scaling with anchors and height constraints for indoor localization of mobile nodes. *Robotics and Autonomous Systems* (2018).

[10] Doherty, Lance et al.. 2001. Convex position estimation in wireless sensor networks. In *Proceedings of the Conference on computer communications.*

[11] Ferrari, Federico et al.. 2011. Efficient Network Flooding and Time Synchronization with Glossy. In *Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks.*

[12] Gallacher, Markus et al.. 2023. InSight: Enabling NLOS classification, error correction, and anchor selection on resource-constrained UWB devices. In *Proceedings of the 2023 International Conference on Embedded Wireless Systems and Networks.*

[13] Garg, Nakul et al.. 2025. Large Network UWB Localization: Algorithms and Implementation. In *Proceedings of the 22nd USENIX Symposium on Networked Systems Design and Implementation.*

[14] George Oikonomou et al.. 2022. The Contiki-NG open source operating system for next generation IoT devices. *SoftwareX* (2022).

[15] Großwindhager, Bernhard et al.. 2018. Enabling runtime adaptation of physical layer settings for dependable uwb communications. In *Proceedings of the 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks".*

[16] Hamer, Michael et al.. 2018. Self-calibrating ultra-wideband network supporting multi-robot localization. *IEEE Access* (2018).

[17] IEEE. 2026. IEEE Draft Standard for Low-Rate Wireless Network. *IEEE P802.15.4ab/D04, December 2025* (2026).

[18] Istomin, Timofei et al.. 2016. Data prediction+ synchronous transmissions= ultra-low power wireless sensor networks. In *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems.*

[19] Istomin, Timofei et al.. 2018. Interference-resilient ultra-low power aperiodic data collection. In *Proceedings of the 17th ACM/IEEE International Conference on Information Processing in Sensor Networks.*

[20] Kabsch, Wolfgang. 1976. A solution for the best rotation to relate two sets of vectors. *Foundations of Crystallography* (1976).

[21] Koledoye, Moses A. 2017. Experiments and samples in Multidimensional Scaling-based localisation. https://github.com/lab-robotics-unipv/mds_experiments. Last accessed: 2025-04-17.

[22] Koledoye, Moses A. et al.. 2017. MDS-based localization with known anchor locations and missing tag-to-tag distances. In *Proceedings of the 22nd IEEE International Conference on Emerging Technologies and Factory Automation.*

[23] Landsiedel, Olaf et al.. 2013. Chaos: Versatile and efficient all-to-all data sharing and in-network processing at scale. In *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems.*

[24] Langendoen, Koen et al.. 2003. Distributed localization in wireless sensor networks: a quantitative comparison. *Computer networks* (2003).

[25] Lobba, Diego et al.. 2020. Concurrent Transmissions for Multi-hop Communication on Ultra-wideband Radios.. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems.*

[26] Neirynck, Dries et al.. 2016. An alternative double-sided two-way ranging method. In *Proceedings of the 13th workshop on positioning, navigation and communications.*

[27] Ridolfi, Matteo et al.. 2021. Self-calibration and collaborative localization for UWB positioning systems: A survey and future research directions. *ACM Computing Surveys (CSUR)* (2021).

[28] Ridolfi, Matteo et al.. 2021. UWB anchor nodes self-calibration in NLOS conditions: A machine learning and adaptive PHY error correction approach. *Wireless Networks* (2021).

[29] Salimpour, Sahar et al.. 2023. Exploiting redundancy for UWB anomaly detection in infrastructure-free multi-robot relative localization. *Frontiers in Robotics and AI* (2023).

[30] Santoro, Luca et al.. 2023. Whereareyou: an uwb relative tracking system for pedestrian using only ranging information. In *Proceedings of the International Instrumentation and Measurement Technology Conference.*

[31] Schmickmann, Florian et al.. 2023. Bring Your Own Positioning System: An Infrastructure-free and Omnidirectional UWB-based Localization Approach. In *Proceedings of the 97th Vehicular Technology Conference.*

[32] Schuh, Maximilian et al.. 2022. First steps in Benchmarking the Performance of heterogeneous Ultra-Wideband Platforms. In *Proceedings of the Workshop on Benchmarking Cyber-Physical Systems and Internet of Things.*

[33] Schuh, Maximilian et al.. 2026. FLock: Source code. https://github.com/LENS-TUGraz/FLOCK.

[34] Schuß, Markus et al.. 2017. A Competition to Push the Dependability of Low-Power Wireless Protocols to the Edge.. In *Proceedings of the Conference on Embedded Networked Sensor Systems.*

[35] Shan, Feng et al.. 2022. Ultra-wideband swarm ranging protocol for dynamic and dense networks. *Transactions on Networking* (2022).

[36] Tiemann, Janis et al.. 2020. CELIDON: Supporting first responders through 3D AOA-based UWB ad-hoc localization. In *Proceedings of the 16th International Conference on Wireless and Mobile Computing, Networking and Communications.*

[37] Trobinger, Matteo et al.. 2020. One flood to route them all: Ultra-fast convergecast of concurrent flows over UWB. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems.*

[38] Van Herbruggen, Ben et al.. 2023. Multihop self-calibration algorithm for ultra-wideband (UWB) anchor node positioning. *IEEE Journal of Indoor and Seamless Positioning and Navigation* (2023).

[39] Čepek, Aleš. 2002. The GNU Gama project-adjustment of geodetic networks. *Acta Polytechnica* (2002).

[40] Ward, Andy et al.. 2002. A new location technique for the active office. *IEEE Personal communications* (2002).

[41] Zignani, Andrew et al.. 2021. Ultra-wideband (UWB) for the IoT–a fine ranging revolution. *ABI Research* (2021).

[42] Zimmerling, Marco et al.. 2020. Synchronous transmissions in low-power wireless: A survey of communication protocols and network services. *ACM Computing Surveys (CSUR)* (2020).