



Technische
Universität
Braunschweig



17. GI/ITG KuVS Fachgespräch Sensornetze


Technical Report

13. & 14. September 2018, Braunschweig

Lars Wolf, Felix Büsching

Institute of Operating Systems and Computer Networks
Technische Universität Braunschweig

<https://doi.org/10.24355/dbbs.o84-201809121401-1>



Technische Universität Braunschweig
Institute of Operating Systems and Computer Networks
Mühlenpfordtstrasse 23
38106 Braunschweig
www.ibr.cs.tu-bs.de

Organization

- Robert Hartung
- Ulf Kulau
- Mareike Schmidt
- Antje Lemke
- Frank Steinberg
- Ulrich Timm
- Yannic Schröder

Program Committee

- Doreen Böhnstedt, TU Darmstadt
- Falko Dressler, Univ. Paderborn
- Horst Hellbrück, FH Lübeck
- Karin Anna Hummel, JKU Linz
- Andreas Reinhardt, TU Clausthal
- Bernd-Christian Renner, TU Hamburg-Harburg
- Thomas C. Schmidt HAW Hamburg
- Bettina Schnor, Univ. Potsdam
- Matthias Waehlich, FU Berlin

FGSN 2018 Preface

It is our great pleasure to welcome you to the 17. GI/ITG KuVS Fachgespräch Sensornetze – FGSN 2018. Welcome also to the city of Braunschweig and the TU Braunschweig, the oldest technical university in Germany. The FGSN workshop series has already a long tradition to serve as forum for the discussion of research on Wireless Sensor Networks, Internet of Things, and other related topics, where especially very research constrained nodes should interact with different types of environments. This tradition will be continued this year with a very interesting program consisting out of 16 paper presentations and 6 demos, all covering highly relevant and recent research issues. We are convinced that this will lead to a highly interactive workshop with lively discussions and intense exchange among all participants. We want to thank all those who contributed to FGSN 2018 and making this workshop possible: the authors for submitting their work to the workshop, the program committee members for reviewing the papers, and the organizing team, especially Felix Büsching, Robert Hartung, and Ulf Kulau, but also all the others from IBR@TU Braunschweig who contributed to the workshop organization. The major value of workshops is to enable discussions among researchers and to allow for the exchange of ideas – which should lead to new thoughts, insights, and results. We hope that the FGSN 2018 workshop can contribute to this and that you will enjoy the event.

Lars Wolf
FGSN 2018 Chair

Contents

Papers	7
IoT Communication Introduced Limitations for High Sampling Rate Applications <i>Silvia Krug and Mattias O’Nils</i>	7
Enabling the Management of IEEE 802.11s Wireless Mesh Networks	11
<i>Michael Rethfeldt, Benjamin Beichler, Peter Danielis, Christian Haubelt and Dirk Timmermann</i>	
A Latency Analysis of IEEE 802.11-based Tactile Wireless Multi-Hop Networks . .	15
<i>Frank Engelhardt, Mesut Güneş</i>	
Packet Merging-driven Tree Construction in Wireless Sensor Networks	19
<i>Aleksandar Ilić and Mario Schölzel</i>	
Managing Swarms of Robots: From Centralized Scheduling to Decentralized Scheduling (work in progress)	23
<i>Daniel Graff and Reinhardt Karnapke</i>	
Evaluation of the 6TiSCH network formation	27
<i>Dario Fanucchi, Barbara Staehle and Rudi Knorr</i>	
Binary Representation of Device Descriptions: CBOR versus RDF HDT	31
<i>Kristina Sahlmann, Alexander Lindemann and Bettina Schnor</i>	
A Case for Chirp Modulation for Low-Power Acoustic Communication in Shallow Waters	35
<i>Jan Heitmann, Fabian Steinmetz and Christian Renner</i>	
Test-Framework zur softwarebasierten Fehlerinjektion, -stimulation und Protokollierung von WSN-Anwendungen	39
<i>Max Frohberg, Sebastian Reinhold, Paul Poppe and Mario Schölzel</i>	
Moving Task Scheduling to Co-Processors in Energy-Harvesting Systems	41
<i>Lars Hanschke, Alexander Sowarka and Christian Renner</i>	

Bayesian Variational Optimization in Sensor Networks	45
<i>Steffen Illium, Thomas Gabor and Thomy Phan</i>	
Extensions at Broadcast Growth Codes	48
<i>Isabel Madeleine Runge and Reiner Kolla</i>	
Using High-Voltage Pulses for Opportunistic Data Transfers in Wireless Sensor Networks	52
<i>Jana Huchtkoetter and Andreas Reinhardt</i>	
A Virtual Reality Multi-Sensor 3d Reconstruction System	56
<i>Markus Friedrich and Kyrill Schmid</i>	
Challenges and Opportunities of Wireless Underground Sensor Networks	60
<i>Idrees Zaman and Anna Förster</i>	
RIZO - RFID basiertes Zeiterfassungssystem für Outdoorsportarten	64
<i>Mario Redinger and Alexander von Bodisco</i>	
Demos	68
Demo: An Ontology-based NETCONF-MQTT Bridge for Sensor Devices in the IoT	68
<i>Kristina Sahlmann, Alexander Lindemann, Thomas Scheffler and Bettina Schnor</i>	
Demo: A Case for Chirp Modulation for Low-Power Acoustic Communication in Shallow Waters	71
<i>Fabian Steinmetz, Jan Heitmann and Christian Renner</i>	
Demo: An Interactive Demonstration of the PotatoScanner, a Mobile DTWSN Node	73
<i>Björn Gernert and Lars Wolf</i>	
Demo: InPhase – 3D Localization in Wireless Sensor Networks	75
<i>Yannic Schröder and Lars Wolf</i>	
Demo: BATS a Flexible System for Automated Encounter Detections	77
<i>Björn Cassens and Rüdiger Kapitza</i>	
Demo: Combining Temperature-Controlled Chambers to Simulate Energy Har- vesting from Thermal Differences	79
<i>Robert Hartung, Sven Pullwitt and Lars C. Wolf</i>	
Demo: Scheduling for Passive Undervolting of Peripheral Components	81
<i>Ulf Kulau, Stephan Friedrichs and Lars Wolf</i>	

IoT Communication Introduced Limitations for High Sampling Rate Applications

Silvia Krug and Mattias O’Nils
Department of Electronics Design
Mid Sweden University
 Sundsvall, Sweden
 {silvia.krug, mattias.onils}@miun.se

Abstract—Networking solutions for the Internet of Things are typically designed for applications that require low data rates and feature rare transmission events. The initial assumption leads to a system design towards minimal data transfers and packet sizes. However, this can become a challenge, if applications require different traffic patterns or cooperative interaction between devices. Applications requiring a high sampling rate to capture the desired phenomenon produce larger amounts of data that need to be transported. In this paper, we present a study highlighting some of the challenging aspects for such applications and how the choice of communication technology can limit both application behavior and network structure.

Index Terms—Internet of Things; High Sampling Rate Applications; Performance Evaluation.

I. INTRODUCTION

Traditional assumptions on the communication in the Internet of Things (IoT) feature a large amount of devices connected each requiring low data rates and infrequent messages. With this assumption as a base several new communication technologies such as LoRa or NB-IoT amongst others were introduced and developed. The goals are always to support many devices and ensure long devices lifetimes. Duty cycles with long sleep phases and low rates with few bit or byte per day and device shall guarantee this.

However, these assumptions result in two consequences that might limit the application spectrum and have an impact on possible deployments. The requirement to send only a few byte per day and sleep otherwise fits to leaf nodes equipped with sensors that provide corresponding sensor characteristics or apply in-node processing to reduce the data amount for communication. Sensor characteristics here imply that only few values are measured and communicated matching the communication technologies limits. This applies to temperature sensors but becomes more difficult in case of other applications that involve high rate sampling or data-intensive sensors such as cameras. For these applications, the “few byte per day” can only be achieved, if processing is done locally at the cost of spending energy for the processing instead [1].

If distributed processing or sensing are required by the application scenario, the focus of few packets per node becomes more challenging and possibly unable to achieve. The same is true, if the network structure requires multi-hop communication and possibly data aggregation or fusion within nodes. In these cases, nodes transport typically higher amounts

of data, either as relayed small packets or exploiting the available packet sizes by combining data into bigger packets. Besides that, nodes communicate more often if larger amounts of data are to be transported, altering their activity profile and thus their energy consumption. For these nodes, minimal packet sizes might not be the best way of handling the traffic and other more traditional technologies might provide a better communication service. However, even if these technologies were not designed for applications there are several requests to still apply them.

In this paper, we present a study for an vibration monitoring sensor application as an example with high rate requirements. After a brief review on the impact of packet sizes in general, we theoretically review the interaction between packet sizes and sampling rates as well as the impact of traffic from neighboring nodes. Besides that, we also perform measurements to further analyze the impact of stack implementation and overhead required to transfer the required data.

II. RELATED WORK

The impact of variable loads on battery-powered IoT nodes is studied in [2], where the authors introduce schemes with variable packet sizes in order to prolong the network lifetime. They effectively show that a data reduction of a few bit can have a significant impact of the node lifetime and suggest to develop custom protocols to avoid the overhead introduced by standardized communication. While this can be an option for isolated networks or networks with translating gateways, it is not an option in heterogeneous environments where nodes might be replaced/added during the system life-cycle. In that case, standardized protocols with a unified communication interface are preferred [3]. How severe the impact of small packets in such a case is, remains however open.

In [4], the authors present a survey on the requirements for real-time analytics of data gathered by IoT devices in order to develop further knowledge on the observed process. Such a processing step can also be integrated into adaptive systems to control the system characteristics based on patterns observed at real-time. However, the authors state that the requirements for a network supporting such analytics are currently not considered by most networking oriented research. This fits to the observation that especially the data flows required for

in-network processing are not supported by some recent IoT communication technologies.

Another crucial role in the effectiveness is the overhead that gets introduced when actually running applications on top of a chosen technology. In order to ensure the interoperability of node, IP-based communication is applied [5] with two additional protocol layers on top of IP and various adaptation layers to enable IP-over-X communication [3]. The stack configuration can have a significant impact on the amount of transmitted data [6]. This does not only include header information of higher layers that reduce the available payload but also robustness mechanisms that require more frequent transmissions in case of errors.

In this paper, we will provide further insights on the effect data flow as well as unusual traffic patterns on an IP-based IoT network. The work therefore extends the existing work by adding a comparison with real-world measurements and a discussion on potential limitations to the application use case.

III. THEORETICAL ANALYSIS OF DATA FLOW OPTIONS

The actual data flow within the network has an impact on how often nodes have to communicate. Figure 1 compares three simple scenarios that represent the general options and can be combined for further more complex scenarios.

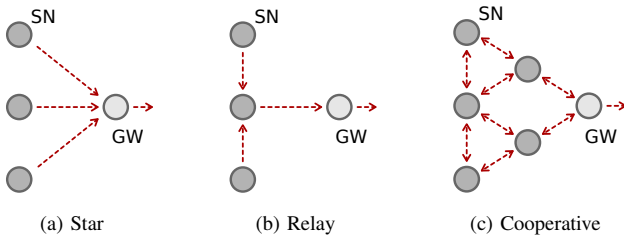


Fig. 1. Comparison of data flow for different scenarios

We consider a the following base scenarios:

Star (cf. Figure 1a) where all sensor nodes (SN) directly communicate with a gateway (GW). The data flow in this case is mainly in a single direction towards that gateway, even if control messages might be required in opposite directions. All nodes handle their own traffic in this case and only have to coordinate the access to the shared medium among neighboring nodes.

Relay (cf. Figure 1b) where the sensor nodes communicate either directly with the gateway if they are within range or use neighboring nodes as relay towards it instead. The direction of the data flow remains towards the gateway but data aggregation or fusion can be possible at intermediate relay nodes. In this scenario, the relay nodes have to handle their own traffic as well as that of their neighbors and potentially spend more effort with processing the received data.

Cooperative (cf. Figure 1c) where all sensor nodes (SN) communicate with their neighbors in order to perform a distributed task. In contrast to the previous cases, the cooperative processing requires that the neighbors exchange

data in both directions and only communicate the result to the gateway. From a data flow perspective this is the most complex scheme that requires coordination between the nodes on application level.

Based on a distributed vibration monitoring system as example application, we first have a look at the expected data amount per sensor node and then analytically review what happens in each of the above scenarios. As example sensor we use the MPU 9250 a 9-degree-of-freedom motion tracking chip that features an accelerometer, a gyroscope and a magnetometer. For vibration monitoring, the acceleration measurements are relevant. These values are provided as a 16 bit sample value for each of the three axes, resulting in a raw data amount of 48 bit per sample. Possible sampling frequencies range from 0.25–1000 Hz depending on the configuration settings. It should be noted that this configuration is based on division and thus the resulting sample rate is not arbitrarily tunable.

Based on the sensor-specific values, we calculated the required data rate for a single sensor node. Figure 2 shows the corresponding results and an additional aggregation step. This figure shows that a single sensor with maximum sample rate is able to utilize the complete theoretical available data rate of 802.15.4 (dashed line), if each sample is send individually. The reason is the high overhead of 48 Byte for the IPv6 and UDP headers in comparison to the relatively small amount of 6 Byte per sample. Using this simple and straight forward approach is therefore possible but not efficient, if the system consists of more than one node.

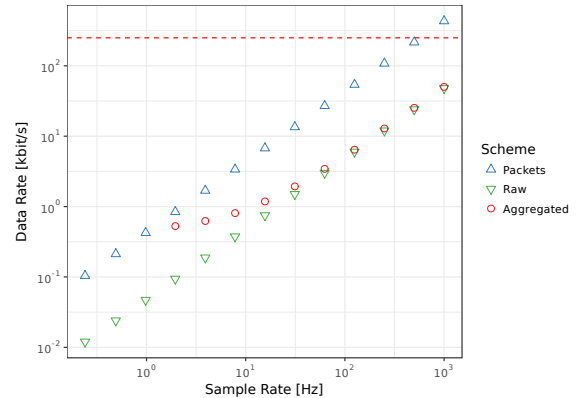


Fig. 2. Impact of data buffering

If we instead buffer the samples in order to aggregate them into as few as possible IPv6 packets per second, the result is much better. To evaluate this, we calculated the resulting data amount for all sample rates that result in more than one sample per second and thus at least one packet. As we assume that at least one packet per second is send, there is some overhead for lower sampling rates, while for high sampling rates the overhead gets significantly smaller almost reaching the data rate requirement of the raw data.

Whether such a buffering scheme is tolerable depends however on the application scenario at hand. One packet per

second might be to slow in case of real-time analysis but can be ok if a long time monitoring is to be performed and the analysis of the data takes place at a later point in time. If time of delivery is not crucial other aggregation schemes are also possible. For example, one could wait until enough application data is gathered to fill one IPv6 MTU. This would result in different sending intervals depending on the chosen sample rate.

Since the goal is to also evaluate possible data flow options towards cooperative distributed processing, the previous results indicate one more thing. Even if we send raw data or buffer it as described, the available theoretical data rate of 802.15.4 is limiting the number of ongoing transmissions and thus the number of neighboring nodes effectively, if we assume multiple sensor nodes producing the same amount of data.

Only 4 full data sets can be transported on the available bandwidth in parallel if the data are produced with the highest sampling frequency of 1 kHz. In the first scenario mentioned above, this corresponds to 4 sensor nodes plus gateway and limiting this number further in the other cases, as the nodes would have to compete for the medium while retransmitting information when acting as relays. Due to this, we assume a maximum sampling rate of 250 Hz for the remainder of the paper. This is an expected behavior showing a possibly limiting factor to IoT applications. To achieve a local cooperative processing, a balance between the local network load and the application requirements is crucial.

Figure 3 shows the corresponding load of a single node, that has to relay data from neighboring nodes assuming a network depth of one hop. Each further hop results in more incoming traffic that has to be handled by the nodes. This includes the aggregation of flows towards a central gateway and potentially further data gathering at each node at each relay, if these nodes participate in the sensing task and do not represent mere infrastructure nodes.

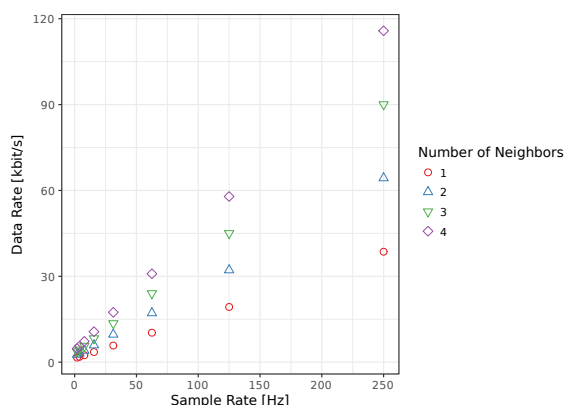


Fig. 3. Impact of relaying traffic from neighboring nodes

The increased traffic in the surrounding of a relay node is using up the available bandwidth fast in all cases. It should be noted, that the calculations here represent an ideal case, where nodes access the medium fairly and no retransmissions or interference from other nodes occur. If that is not the case,

even lower sampling rates are available for the application or other networking technologies should be evaluated.

IV. MEASUREMENT CAMPAIGN

To evaluate the practical impact of the considerations above, we performed measurements using real hardware. We performed our experiments using the OpenMote [7] platform with RIOT [8] operating system as base. As networking stack in RIOT, we used the default generic networking stack (GNRC) [9]. On top of that, we used a custom UDP application to send and receive different application payloads that is based on the `gnrc_networking` example application in RIOT. The application has two functions:

- send single UDP packets with increasing payload sizes from 1 to 1232 bytes in steps of 1 byte and
- send a user selected data amount

The first function allows us to measure the impact of the overhead and the maximum possible payload up to the maximum transmission unit (MTU) of 6LoWPAN since its RIOT implementation does not support automatic fragmentation of IPv6 packets [9]. The second function is used to send an arbitrary data amount by manually fragmenting the data into suitable payload chunks. This is therefore an option to analyze the stack behavior in case of large data amounts.

In both functions, multiple packets are sent with each configured payload size in order to account for variations in the channel. For each packet, we record timestamps for the initial packet generation as well as the start and the end of the transmission and log these values to the UART console. Based on these timestamps we can calculate how long a transmission of single packets as well as the whole data amount takes and derive the actual throughput from that.

Surprisingly, with the default configuration of GNRC, it was not possible to successfully receive UDP packets with a payload bigger than 81 Byte. While this is not a problem for low rate applications that actually send few bytes only, it becomes a problem for applications that have to handle higher amounts of data efficiently. For these applications the observed limitation results in the need for manual packet fragmentation. After adapting the GNRC buffer size to fit one 6LoWPAN MTU according to the description in [9], this problem was solved. Figure 4 shows the impact of both schemes.

The limitation of the application payload to 81 Byte introduces severe additional overhead because the same data amount requires much more packets in order to be transferred, each again with the 48 Byte header overhead for UDP and IPv6. Besides this additional data, the processing overhead is increased too, due to necessary packet preparation. Our results show another important aspect of the system design: the appropriate and verified configuration of the software stack used. This is especially true for applications with traffic patterns that vary from the typical design assumptions.

As for the required instantaneous application throughput for a single node, the measurements with incremental increase of the payload size give valuable insights, too. Figure 5 shows the throughput variation over the analyzed payload range.



Fig. 4. Measured application throughput depending on payload size setting

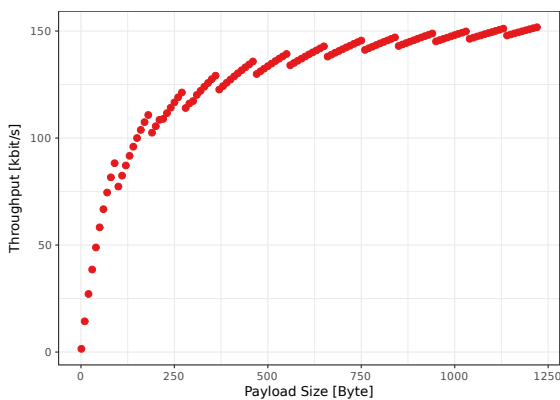


Fig. 5. Throughput of a single sending node over the payload range

The observed gaps correspond quite well to the underlying fragmentation of the 6LoWPAN packet into corresponding 802.15.4 MAC frames. Whenever an additional frame is required to transport the requested data amount, the throughput drops slightly, due to the increased header overhead. A single node would require up to 150 kbit/s with a delay of 100 ms per packet for the maximum packet size. According to this result, 9 to 10 such packets could be sent in the surrounding of a node within one second. These results clearly show that low rate networks are not suitable for a cooperative vibration monitoring task. While this was clear from the beginning for new technologies like LoRa or NB-IoT, it was not expected to be so severe for other WSN technologies.

Even 6LoWPAN networks can be limiting the application because the available data rate limits the sampling frequency or data production rate, if no in-node preprocessing is done to reduce the data amount. For the given use case, the goal is to sample with a high rate well above 100 Hz. This is still possible but results in deployments with either few nodes in the network and a direct flow towards the gateway. Distributed processing among the nodes or coordination/adaptation between them becomes difficult to achieve.

These experiments show, that the usage of 802.15.4 in combination with 6LoWPAN is challenging for high rate

applications. However, it is possible to use the technologies, if the application needs and network configuration is balanced accordingly. If higher performance is required, alternative communication technologies have to be evaluated and developed.

V. CONCLUSIONS

In this paper, we first reviewed the impact of high sampling frequencies on the amount of data that a node has to handle in a vibration monitoring task. The study also includes a comparison with options to distribute the data and adds measurement results to complement the theoretical insights.

Our results show that applications with high sampling rates or otherwise large amounts of data are posing several challenges to the network. 802.15.4 as one widespread technology with a higher data rate supports only few nodes in such a system or requires the application to limit its sampling frequency. If that is not an option, only in-node processing is an option and the network should be designed as a star network with a data flow to the gateway.

In the future, we plan to extend this work by exploiting further networking technologies such as WiFi and Bluetooth Low Energy as well as explore the impact of distributed processing within the network.

REFERENCES

- [1] M. Imran, K. Shahzad, N. Ahmad, M. ONils, N. Lawal, and B. Oelmann, "Energy-Efficient SRAM FPGA-Based Wireless Vision Sensor Node: SENTIOF-CAM," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 12, pp. 2132–2143, 2014.
- [2] M. D. Prieto, B. Martínez, M. Monton, I. V. Guillen, X. V. Guillen, and J. A. Moreno, "Balancing power consumption in IoT devices by using variable packet size," in *8th International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*. IEEE, 2014.
- [3] C. Gomez, J. Paradells, C. Bormann, and J. Crowcroft, "From 6LoWPAN to 6Lo: Expanding the Universe of IPv6-Supported Technologies for the Internet of Things," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 148–155, 2017.
- [4] S. Verma, Y. Kawamoto, Z. M. Fadlullah, H. Nishiyama, and N. Kato, "A survey on network methodologies for real-time analytics of massive IoT data and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1457–1477, 2017.
- [5] T. Watteyne, V. Handziski, X. Vilajosana, S. Duquennoy, O. Hahm, E. Baccelli, and A. Wolisz, "Industrial Wireless IP-Based Cyber-Physical Systems," *Proceedings of the IEEE*, vol. 104, no. 5, pp. 1025–1038, 2016.
- [6] S. Krug, A. Schreiber, and M. Rink, "Poster: Beyond Static Sending Intervals for Sensor Node Energy Estimation," in *12th Workshop on Challenged Networks (CHANTS)*. ACM, 2017.
- [7] X. Vilajosana, P. Tuset, T. Watteyne, and K. Pister, "OpenMote: open-source prototyping platform for the industrial IoT," in *International Conference on Ad Hoc Networks*. Springer, 2015.
- [8] E. Baccelli, O. Hahm, M. Gunes, M. Wahlisch, and T. C. Schmidt, "RIOT OS: Towards an OS for the Internet of Things," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2013.
- [9] M. Lenders, "Analysis and Comparison of Embedded Network Stacks," Master's thesis, Freie Universität Berlin, 2016.

Enabling the Management of IEEE 802.11s Wireless Mesh Networks

Michael Rethfeldt, Benjamin Beichler, Peter Danielis, Christian Haubelt, Dirk Timmermann
Institute of Applied Microelectronics and Computer Engineering, University of Rostock, Germany
Email: michael.rethfeldt@uni-rostock.de

Abstract—The broad availability of WLAN-capable off-the-shelf hardware lets WLAN mesh networks appear as promising technology for future distributed wireless applications. Featuring automatic device discovery, interconnection and routing, they provide a higher scalability, flexibility, and robustness compared to common centralized WLAN infrastructures. Besides these advantages, characteristics such as variable network topologies and link qualities imply new technical challenges for administration and real-world operation. Adopted in late 2011, IEEE 802.11s appears as new WLAN standard amendment, enabling vendor-independent mesh networks based on the widespread WLAN technology. However, network monitoring and management fall out of the standardization scope and are therefore not specified. In this paper, we present a novel 802.11s management solution based on the SNMP protocol. It covers dynamic mesh bootstrapping, error recovery, status monitoring and remote configuration. The presented solution was implemented and evaluated in a real-world testbed comprising more than 10 mesh nodes.

Index Terms—WLAN; Wireless Mesh Network; IEEE 802.11s; Network Management; SNMP; Linux; open80211s; mac80211

I. INTRODUCTION

Both the growing variety and affordability of mobile consumer devices serve as a catalyst for next-generation wireless services and applications. The vision of the “Internet of Things” (IoT) is characterized by the seamless integration of ambient embedded systems for user assistance [1]. Highly cooperative ensembles are built from a broad spectrum of devices such as sensors, actors, smart displays, and especially wearables like smart phones or tablets. This leads to complex networks that can provide distributed information services, e.g., in public places or smart offices. Further applications include home and building automation or the monitoring of industrial facilities. The widely-supported IEEE 802.11 WLAN (Wireless Local Area Network) standard family [2] is already omnipresent in today’s home and office environments. Nevertheless, the currently prevalent centralized infrastructure mode is not well-suited for future applications in terms of scalability, flexibility, and robustness [3]. The network comprises a central access point (AP) and multiple stations (STAs) in direct radio range to the AP. The AP manages data forwarding between the associated STAs and therefore acts as single point of failure. Network extension is commonly achieved by a wired “Distribution System” (DS), e.g., via Ethernet. This leads to costly deployment if good coverage is needed.

In contrast, decentralized WLAN mesh networks are characterized by a flexible, scalable, and fail-safe topology [3]. Distributed mesh routing is handled by all network nodes.

Neighbors within radio range (peers) automatically interconnect and establish paths to selected targets. Thus, the network becomes more robust to changes in link availability and quality. An easy and economic network extension is simply possible by bringing in additional mesh nodes.

The higher dynamics and complexity of WLAN mesh networks put many challenges on monitoring and management, as no central AP is keeping track of the network status and associated devices. Moreover, all nodes must be initialized properly, operate on the same wireless channel, and share a common mesh configuration. In general, many manual configuration steps are required on each device. This is not suitable for a large number of nodes, and especially for complex mesh topologies, such as multi-radio / multi-channel setups [4]. Thus, mesh bootstrapping has to be performed in a distributed fashion where every node is equipped with self-configuration capabilities. Although a self-forming mesh network could already run unattended, the limited scope per node inherently limits optimization potentials. To regain a global network view, it is necessary to collect status data of all nodes [4]. Based on these information, it becomes possible to identify weak regions and misconfiguration, and to derive suitable optimization steps. Thereby, it is desirable to enrich a mesh node with monitoring and management capabilities. Furthermore, devices may not provide any user interface or may not be accessible when mounted on walls or roof tops. Therefore, it is necessary to provide remote control functionality to ensure the practical operation and maintainability of WLAN mesh networks. Consequently, we have developed a centralized management approach with a dedicated device role for remote monitoring and configuration. Dynamic network bootstrapping and error recovery are already performed in a decentralized manner by all mesh nodes.

II. TECHNOLOGICAL BASIS

A. IEEE 802.11s

In September 2011, IEEE 802.11s was ratified as the first industry WLAN mesh standard [2], [5]. As an amendment to the existing 802.11 specification, all mesh functionality is fully integrated into the existing MAC layer while the underlying physical layer (802.11 a/b/g/n/ac) remains untouched. Already existing MAC-layer frame types have been extended to enable mesh functionality such as automatic peering and routing (path selection). By reusing the existing MAC mechanisms, they are expected to perform with less overhead, compared to earlier, incompatible mesh routing protocols. The mesh

topology becomes invisible to all higher layers, which avoids any modifications above the WLAN specification. Next to simple STAs with mesh capability, *Mesh Points* (MP), 802.11s also defines specialized nodes that act as gateways to external networks (*Mesh Portal* - MPP) or as AP for legacy devices (*Mesh Access Point* - MAP). Figure 1 shows an example 802.11s mesh topology.

To ensure interoperability, the 802.11s standard specifies a mandatory basic mesh profile. It consists of the *Hybrid Wireless Mesh Protocol* (HWMP) for path selection and the radio-aware *Airtime Link Metric* (ALM). ALM represents costs for choosing a specific path in the mesh network by considering technology parameters of the physical layer and the wireless medium. Every mesh node is solely aware of its direct peers and nodes in multi-hop distance, to which communication has been explicitly initiated, e.g., on application level. A node's path table exclusively contains forwarding rules to target nodes over suitable next-hop peers, which are continuously updated. To every target node, only the best rule is kept, represented by the smallest ALM cost. Altogether, 802.11s dictates only a minimal mandatory mesh feature set. The use of HWMP inherits a limited network view on each node which renders network-wide management a difficult task. The mesh standard permits the use of vendor-specific path selection protocols that potentially entail a global network view per node. However, this approach would compromise overall interoperability. Thus, further management functions have to be implemented as separate solution.

B. open80211s

The project *open80211s* [6] is a reference implementation of 802.11s for the Linux platform and used as basis for our solution. It is currently the most advanced 802.11s implementation and already satisfies all mandatory and various optional parts of the standard. The program code of *open80211s* is integral part of the WLAN stack inside the mainline Linux kernel.

C. SNMP

The *Simple Network Management Protocol* (SNMP) has become a de facto standard for the management of IP-based LANs [7]. Its first version was specified in 1988 by the IETF. The latest iteration *SNMPv3*, released in 2002, was

extended by complex security mechanisms. SNMP follows a client/server model consisting of manager (client) and agent (server). The SNMP agents, running on every network device, provide their local status information as *Managed Objects* (MOs). MOs are structured in a hierarchical *Management Information Base* (MIB). Every MO can be uniquely addressed by its *Object Identifier* (OID). Many standardized and enterprise MIBs are already available. With the reserved sub-tree "experimental", it is possible to define own MIB extensions, e.g., for research projects. Communication between client and agent is bound to UDP transport by default and follows a simple request-response principle. SNMP defines message types to request (GET), write (SET), and publish (TRAP) MO data, as well as numerous error codes to identify problems. MOs and data types are specified using ASN.1-based syntax rules. For transport, a binary encoding is used. Considering its wide acceptance, large set of features, and lightweight encoding, SNMP appears as a well-suited protocol also for the use in WLAN mesh networks.

III. RELATED WORK

Several management solutions have been proposed for IP-layer WLAN mesh protocols. Many approaches extended SNMP, supported it by proxy, or integrated it directly. As shown in earlier publications [8], [9], SNMP has been successfully applied in common WLAN infrastructures. Nevertheless, there is still a lack of solutions with particular focus on 802.11s and its characteristics. The UAVnet project [10] is the only 802.11s testbed we know of to explicitly include management functionality. The network is deployed by unmanned aerial vehicles (UAV), forming a wireless relay for disaster scenarios. Administration is based on the ADAM framework. It supports firmware deployment, logging per node, time synchronization, link loss recovery and provides a web interface to adjust device settings. Nevertheless, focus is not on mesh management and optimization but mainly on node initialization and flight control. ANMP [11], Guerrilla [12] and MannaNMP [13] present SNMP-based management of ad-hoc networks. Modified SNMP messages and self-defined MIB extensions are used for node monitoring. These works further describe concepts of hierarchical network clustering. Prabhu [14] describes a hierarchical concept for SNMP usage in ad-hoc networks proposing a division of specialized master- and sub-agents for different management tasks.

IV. 802.11s MESH MANAGEMENT SOLUTION

We present a management solution specifically designed for 802.11s networks that addresses bootstrapping, monitoring, and remote configuration. This includes MAC- and IP-layer auto-configuration to ensure node availability for further management tasks. We further provide autonomous error recovery to re-initialize a node after misconfiguration or failure. In contrast to link state routing protocols, HWMP induces a limited mesh topology knowledge per node. Thus, network monitoring is implemented above 802.11s to regain a global scope. Furthermore, standard-specific features (link blocking, path modification, etc.) are made remotely accessible.

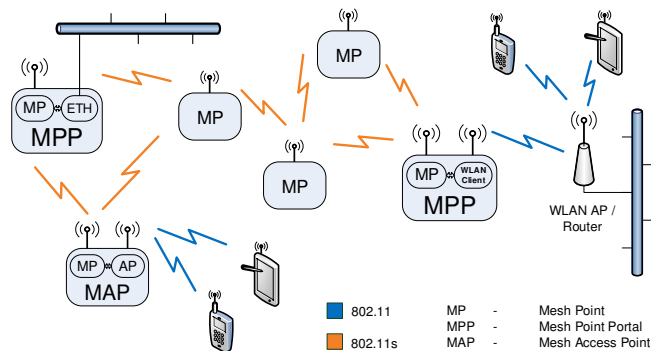


Fig. 1. Example 802.11s mesh topology

Our framework is realized as Java application on top of *open80211s* for Linux. Following a modular architecture, it comprises a larger platform-independent and smaller platform-dependent part. The latter serves as adapter for the underlying OS and can be easily replaced for future 802.11s implementations. Relying on SNMP, it provides the client-side *Mesh Manager* and server-side *Mesh Agent*. We use *SNMP4J* [15] to implement SNMP core functionality. Mesh status data and configuration interfaces are described as MIB extension. This way, interoperability with other SNMP-supporting management tools is ensured.

As shown in Figure 2, the platform-specific part encapsulates user space tools for WLAN interface setup (*iw*), bridging (*brctl*), AP authentication (*hostapd*), IP and routing configuration (*ip*, *iptables*), DHCP functionality (*dnsmasq*, *dhclient*), and time synchronization (*ntp*). Data extracted from the OS include general information like the list of current WLAN adapters, virtual interfaces, or IP routes as well as 802.11s-specific information. The “Peer List” and “Mesh Path List” hold status information for every link entry and path forwarding rule. While any link entry includes status, RSSI, sent and received bytes, or frame retransmission and failure count, any forwarding rule consists of next hop node, target node, and ALM metric [6]. Aggregation of the distributed link and path information allows it to derive a global network scope. Moreover, by considering information such as link bandwidth, RSSI, or ALM, it is possible to infer relative node distance and spatial mesh topology.

A. Mesh Agent

The *Mesh Agent* core is built as state machine to handle dynamic initialization and self-configuration. This includes a detection of errors such as failing WLAN adapters and IP address loss. To be reachable via SNMP, at least one mesh interface must be connected at any time, both on 802.11s and IP level. Furthermore, a mesh node must not be isolated. Thus, its peer list is continuously checked for active links. Fall-back routines are triggered automatically on occurrence

of a *noNeighborTimeout* or *noIPAddrTimeout* to restart necessary initialization steps. A *Mesh Agent* traverses the states *INIT* (initializing mesh interface), *SCANNING* (scanning for surrounding mesh networks), *CONNECTED_L2* (connected on 802.11s MAC layer), and *CONNECTED_L3* (fully connected on IP level with SNMP agent running). Nodes automatically join the found mesh network with highest signal strength. If scan results are empty, a default profile is used. A DHCP client is started for IP address retrieval from a Manager, acting as DHCP server. When fully bootstrapped, an SNMP agent provides its status information (extracted from the OS) and remote configuration interfaces, specified as MIB extension. For Object Identifier (OID) assignment, we chose a free sub-tree in the *experimental* MIB branch (OID .1.3.6.1.3). We put “Status Data Objects” in **.experimental.1234.1* and “Configuration Objects” in **.experimental.1234.2*.

B. Mesh Manager

The *Mesh Manager* role features the complementary SNMP client and DHCP server side to the Agent, as well as a Java GUI for status data visualization and remote configuration. Having joined a mesh network, a Manager checks if it is already managed, indicated by a successful DHCP request. Since multiple Managers are not supported yet, this is repeated until an unmanaged network is found. On success, the Manager periodically sends SNMP GET requests for all “Status Data Objects” to all available Agents. Long-term database storage enables the calculation of statistics and derivation of optimization steps out of the recorded network history. Remote configuration is performed by modifying a Mesh Agent’s “Configuration Objects” through SNMP SET requests. Possible commands include wireless channel selection, node type changes (MP, MPP, MAP), path modifications, peer blocking, and the tuning of further 802.11s-specific parameters related to path selection or link establishment.

V. REAL-WORLD TESTBED EVALUATION

We established a single-channel real-world 802.11s mesh testbed to show the practical suitability of our management solution. It comprises 10 ARM-based devices (1 x FOX Board G20, 7 x Raspberry Pi, 2 x Pandaboard [16]–[18]) as Agents and 1 notebook as Manager ((2 x 1,3 GHz Pentium SU4100, 4 GB RAM). All devices run a Linux OS (Debian 7, Ubuntu 12.04) with kernel v3.16-1 and use an 802.11g WLAN USB stick depending on the *rt2800usb* driver [19], [20].

First, we determined the maximum achievable UDP data rate for different hop counts to classify monitoring overhead. Only the notebook and the Raspberry Pis were used, all operating on channel 1 using default mesh parameters [6]. We used the “Peer Link Block” option to blacklist certain peers of a node and create a multi-hop path. For every path length, 10 UDP throughput measurements between Manager and last Agent were performed using *iperf* v2.0.5 [21]. Measurements were run for 10 seconds with a target rate set above the 802.11g maximum gross rate 54 MBit/s. For each path length, results were averaged, as shown in Figure 3. For 1 hop, the UDP net data rate was 23.7 MBit/s. After 2 hops, it dropped

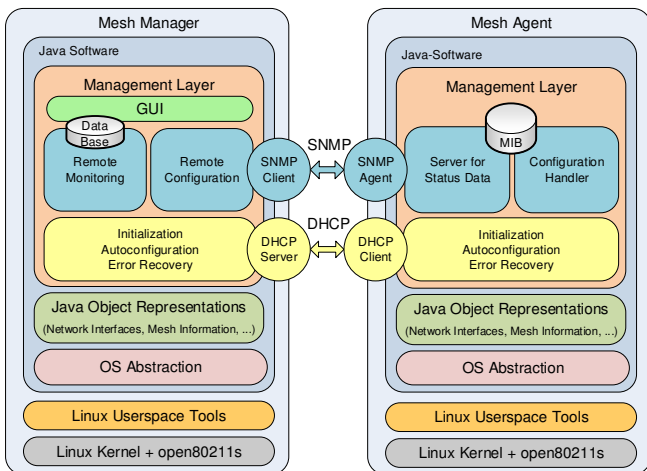


Fig. 2. Architecture of mesh management solution

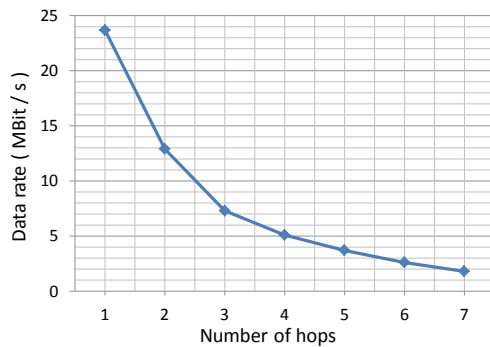


Fig. 3. Multi-hop UDP throughput

to 12.9 MBit/s. For 7 hops, an average of only 1.7 MBit/s was measured. This trend could be expected, as every hop requires channel access and transmission time.

As a second step, we evaluated the cyclic SNMP monitoring overhead for 1 to 10 active Agents in 1-hop distance to the Manager. SNMP traffic was captured using Wireshark v1.10 (UDP datagrams without header) [22]. Separately, 802.11 control frames were captured for direct comparison. The query interval was set to 15 seconds. In each cycle, 10 MO tables were collected from each node. 40 cycles were averaged for every node count and then normalized to 1 cycle. Figure 4 shows the results. As expected, monitoring overhead scales linearly with node count and query interval. For 10 active Agents, only an average of 150 kB data is generated per query cycle. For an interval of 15 seconds and 10 Agents, the average rate is 80 kBit/s. Compared to the achievable single-hop UDP data rate (23.7 MBit/s) in the unmonitored network, this is an overhead of only 0.3 %.

In contrast to the single-hop setup, in a multi-hop scenario SNMP traffic will be forwarded by intermediary nodes and induce a higher overhead. As a next step, we will therefore investigate multi-hop scenarios and explore strategies to facilitate scalability of our solution. Orthogonal WLAN channels can be employed to increase throughput by reducing medium contention and interference. Grouping of nodes into monitoring clusters can help to keep communication paths short and reduce forwarding overhead. Additionally, SNMP supports TRAP messages for notifications that can be used to trigger a query cycle only on certain events.

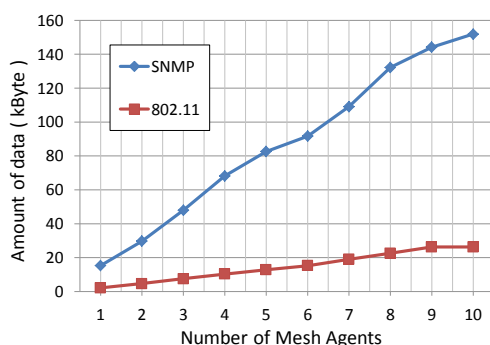


Fig. 4. SNMP overhead depending on node count

VI. CONCLUSION

In this paper we present a Java-based management framework, tailor-made for 802.11s WLAN mesh networks. Our solution provides self-configuration and error recovery capabilities on top of 802.11s and further implements SNMP-based monitoring and remote configuration of the distributed nodes. Thereby, it regains a global network view that is otherwise lost due to a limited network scope, induced by HWMP path selection. Our 802.11s-specific MIB extension already covers the essential status data and configuration functions, needed for practical mesh operation and optimization. For our framework to become as fail-safe as the underlying 802.11s network, in a next step the currently centralized Manager role will become decentralized. In future research we will use our framework to explore cross-layer optimization strategies within the boundaries of the 802.11s standard.

REFERENCES

- [1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.
- [2] "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 802.11-2012 (Revision of IEEE Std 802.11-2007)*, 2012.
- [3] M. Portmann and A. A. Pirzada, "Wireless mesh networks for public safety and crisis management applications," *Internet Computing, IEEE*, vol. 12, no. 1, pp. 18–25, 2008.
- [4] J. L. Duarte, D. G. Passos, R. L. Valle, E. Oliveira, D. C. Muchaluat-Saade, and C. V. N. de Albuquerque, "Management Issues on Wireless Mesh Networks," in *LANOMS*. IEEE, 2007, pp. 8–19.
- [5] G. R. Hiertz, D. Denteneer, S. Max, R. Taori, J. Cardona, L. Berlemann, and B. Walke, "IEEE 802.11s: The WLAN Mesh Standard," *Wireless Commun.*, vol. 17, no. 1, pp. 104–111, Feb. 2010.
- [6] "open80211s." [Online]. Available: <http://open80211s.org>
- [7] R. Presuhn, "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)," RFC 3416 (INTERNET STANDARD). Internet Engineering Task Force, Dec. 2002. [Online]. Available: <http://www.ietf.org/rfc/rfc3416.txt>
- [8] R. Johnson, *Evaluating the use of SNMP as a wireless network monitoring tool for IEEE 802.11 wireless networks*. ProQuest/UMI, 2011.
- [9] C. M. Vancea and V. Dobrota, "SNMP Agent for WLAN networks," *8th RoEduNet International Conference 'Networking in Education and Research', Galati, Romania*, 2009.
- [10] S. Morgenthaler, T. Braun, Z. Zhao, T. Staub, and M. Anwender, "UAVnet: A mobile wireless mesh network using unmanned aerial vehicles," in *Globecom Workshops*. IEEE, 2012, pp. 1603–1608.
- [11] W. Chen, N. Jain, and S. Singh, "ANMP: Ad hoc network management protocol," *Selected Areas in Communications, IEEE Journal on*, vol. 17, no. 8, pp. 1506–1531, 1999.
- [12] C.-C. Shen, C. Jaikao, C. Srisathapornphat, and Z. Huang, "The Guerrilla management architecture for ad hoc networks," in *MILCOM 2002. Proceedings*, vol. 1. IEEE, 2002, pp. 467–472.
- [13] F. A. Silva, L. B. Ruiz, T. R. M. Braga, J. M. S. Nogueira, and A. A. F. Loureiro, "Defining a wireless sensor network management protocol," in *LANOMS*, 2005, pp. 39–50.
- [14] V. Prabhu H, "Master Subagent Based Architecture to Monitor and Manage Nodes in Mobile Ad-Hoc Networks," *International Journal of Engineering Research and Applications (IJERA)*, vol. 2, no. 3, pp. 1461–1465, May 2012.
- [15] "SNMP4J." [Online]. Available: <http://www.snmp4j.org/>
- [16] "ACME Systems." [Online]. Available: <http://www.acmesystems.it/>
- [17] "Raspberry Pi." [Online]. Available: <http://www.raspberrypi.org/>
- [18] "PandaBoard." [Online]. Available: <https://elinux.org/PandaBoard>
- [19] "Buffalo AirStation N150 WLI-UC-GNM." [Online]. Available: <https://www.buffalotech.com/products/airstation-n150-wireless-usb-adapter>
- [20] "rt2x00 Project." [Online]. Available: <http://rt2x00.serialmonkey.com/>
- [21] "iperf." [Online]. Available: <https://iperf.fr/>
- [22] "Wireshark." [Online]. Available: <http://www.wireshark.org/>

A Latency Analysis of IEEE 802.11-based Tactile Wireless Multi-Hop Networks

Frank Engelhardt, Mesut Güneş
 Communication and Networked Systems (ComSys)
 Faculty of Computer Science
 Otto-von-Guericke University Magdeburg
 Universitätsplatz 2, 39106 Magdeburg, Germany
 {fengelha, guenes}@ovgu.de

Abstract—The Tactile Internet idea has recently gained interest as it closes the gap towards carrying the full set of human senses across communication networks. A prerequisite for this vision is the availability of low-latency wireless networks. Wireless Multi-Hop Networks (WMHNs) are a beneficial extension of cellular networks, like 5G, to achieve this goal, since they extend connectivity in a flexible and more dynamic manner. In this paper, we study the effects that WMHN can have on the latency, which is one of the main requirements of haptic communication. We specify certain bounds that the network has to fulfill. Moreover, we present a linear model for the delay of haptic flows in WMHN.

Index Terms—Tactile Internet, Wireless Multi-Hop Networks, Low-Latency Communication

I. INTRODUCTION

The idea of the Tactile Internet [1] is to enable low-latency, high-resolution and highly available services for human-to-machine and machine-to-machine interaction. Through the availability of a respective core network infrastructure, applications can exchange data at the high rates necessary for tactile sensors and actuators. Packet rates of 1 kHz at a maximum latency of 1 ms are required to support those applications, like for example Networked Control Systems (NCSs), teleoperation, telepresence and Haptic Communication. Tactile Internet Applications currently evolve around the upcoming mobile communication standard 5G, which is a promising candidate to fulfill these requirements. Since 5G will support Device-to-Device (D2D) functionality, it will introduce a multi-hop characteristic in addition to its otherwise cellular nature. However, this fact is rarely covered in literature.

The focused application in this paper is a wireless multi-hop teleoperation system for a mobile robot (see Figure 1), which is a good example of a Tactile Internet application [2]. A human user remotely controls a robot from a long distance without direct eye contact, relating only on the feedback returned from the robot over the network. Low latency as well as a continuous availability is a crucial demand towards the network to support the operation. We can easily infer that with WMHN-based teleoperation systems, certain negative effects appear: First, transmission delay multiplies with the hop count, effectively constraining the path length due to the given maximum-latency limit of 1 ms. Second, all routers will interfere with each other, introducing more contention, such that the system may even not have linear behavior. We are

therefore interested in how many hops can a network actually support in this specific application and base our analysis on the general h -hop network shown in Figure 1.

In this paper, we want to elaborate on the aforementioned challenges of tactile applications running on current IEEE 802.11 [3] WMHN technology. Due to the early stage of 5G development at the time, IEEE 802.11 is a good reference technology in terms of throughput, spectrum usage, flexibility and scalability. It also includes mesh-networking support.

II. RELATED WORK

The idea of a Tactile Internet combines knowledge from many research fields. Effects of the communication channel on robot teleoperation have been investigated very early by Sheridan and Verplank [4]. Models tied to the needs of a modern Tactile Internet have been proposed by numerous works of Steinbach et al. [5].

With modern mobile communication technology, especially 5G, the ideas of transmitting user experience have gained a new push [1]. Because the core networks are well understood nowadays, modeling can start to integrate the wireless edge [2]. Recent developments and upcoming standards of ultra-low latency networks integrate core networks and wireless edge networks. Services requiring ultra-low delay, jitter and packet error rate can be shaped using rigorous cross-layer approaches throughout the internet [6].

In our work we focus on applications that do not require strong jitter bounds and thus are of soft real-time nature. We therefore focus on solutions for IEEE 802.11-based WMHNs.

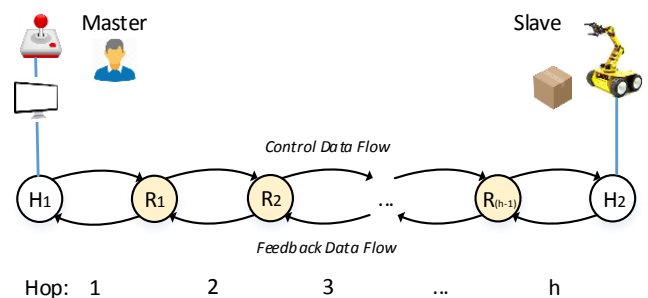


Figure 1: A Tactile Wireless Multi-Hop Network

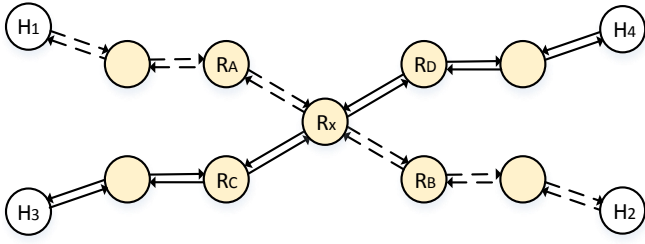


Figure 2: A Tactile Wireless Multi-Hop Network with two applications $H_1 \leftrightarrow H_2$, $H_3 \leftrightarrow H_4$. The two data flows interfere at router R_x

III. CHALLENGES WITH TACTILE APPLICATIONS IN IEEE 802.11 WMHN

Tactile Internet applications generally have two requirements towards the network: They need an end-to-end latency ≤ 1 ms and a packet rate supported of 1 kHz in both directions [1]. The packet size needed for force control and feedback is however very small; in most cases 6-degree-of-freedom vectors of control or force values, respectively, need to be exchanged. The packet rate requirement is therefore easily met with IEEE 802.11 due to its high throughput. But the latency requirement is a very hard criterion. Latency typically results from processing time, interference, congestion, queueing delay and path length. We focus here on the two latter aspects.

A. Single Tactile Application Networks

We first assume a WMHN with only one tactile application communicating between hosts H_1 , H_2 as shown in Figure 1. Without limiting generality, we can ignore all routers that are not part of the route $H_1 \leftrightarrow H_2$, thus we have a linear network with H_1 , H_2 at both ends and a set of $h - 1$ routers in between. End-to-end latency d_{e2e} is proportional to the hop count h , as long as the medium utilization is low enough to support the transmission of the two 1 kHz data flows along both directions. We therefore have a hop count limit when we require $d_{e2e} \leq 1$ ms.

B. Multiple Tactile Application Networks

When more than one tactile application is present within one network, the proportionality $d_{e2e} \sim h$ is violated by intersections between the flows. Figure 2 shows such a network with two flows present which intersect at a router R_x . (When we have a network with two flows that do not intersect, we can consider these as two independent networks, and thus as two separate single tactile application networks.) Router R_x can only send one packet at a time, but has to relay four packets (potentially) simultaneously: $H_1 \rightarrow H_2$, $H_2 \rightarrow H_1$, $H_3 \rightarrow H_4$, $H_4 \rightarrow H_3$. Each of these needs two transmissions, for example $R_A \rightarrow R_x \rightarrow R_B$ for transmitting the direction $H_1 \rightarrow H_2$. The per-flow end-to-end delay again depends on the hop count, but also on the load of router R_x . This load results in queueing delay which itself depends on the number

of flows n intersecting at R_x . Our idea is to calibrate a linear model

$$d_{e2e} = k_1 \cdot h + k_2 \cdot n. \quad (1)$$

We show the average queueing delay can be approximated by a linear upper bound $k_2 \cdot n$ for small n in section V.

IV. THE TACTILE COORDINATION FUNCTION (TCF)

Frame scheduling is a crucial problem in all WMHN when it comes to delay reduction. But at the same time, WMHNs need to maintain their flexibility, scalability and self-X properties as well. This encourages us to solve the scheduling problem in a way that offers dynamic behavior and reduces overhead of dynamic configuration and adaptation as much as possible. The present wireless technology IEEE 802.11 [3] uses the Enhanced Distributed Channel Access (EDCA) scheduling scheme that meets these requirements, but needs adaptation to support tactile data flow. We summarize our findings in this section.

A. EDCA Overview

The IEEE 802.11 EDCA mechanism categorizes traffic into four different Access Categories (ACs): Background (AC_BK), Best Effort (AC_BE), Video (AC_VI) and Voice (AC_VO). Each AC requires different Quality of Service (QoS)-parameters from the network, or, more specific, have specific delay requirements. The ACs are therefore mapped to a user priority (UP), which is an integer value. The bigger the value, the higher the priority. The EDCA scheduler sorts traffic therefore into different queues. Prioritization is static in the queues, i.e. the next frame to send is always the first frame of the highest-priority queue that is not empty.

On the medium, prioritization of frames is regulated within the coordination function by specifying different waiting times for the frames before they are sent. The Hybrid Coordination Function (HCF) of IEEE 802.11, which is a CSMA/CD variant, adjusts waiting times using both the Inter-Frame Space (IFS) and Contention Window (CW) mechanisms. The waiting time is defined as $IFS + aSlotTime \cdot cw$, where the contention window cw is a random number within the range $[cw_{min}, cw_{max}]$.

B. EDCA Tactile Control Function Supplement

Since the tactile data flows need lowest latencies, introduction of a new AC for tactile data with highest priority is obligatory. We declare the Tactile Control (AC_TC) Access Category and adjust the corresponding IFS and cw parameters such that prioritization is guaranteed. Parameters are shown in Table I. We leave the same settings for cw as for AC_VO but reduce the IFS to one slot and increase the UP of the queue. Formally, this suffices to enforce the highest priority as long as the system does not enter the backoff mechanism.

With the given prioritization of tactile data, we can analyze the situation depicted in Figure 2. The intersection router R_x appears as the bottleneck of the overall system and its scheduling behavior determines the overall delay. We show the mechanics in Figure 3. Router R_x needs to be busy as long as all adjacent routers have transmitted their packets. Since the application hosts $H1-H4$ send with a constant rate

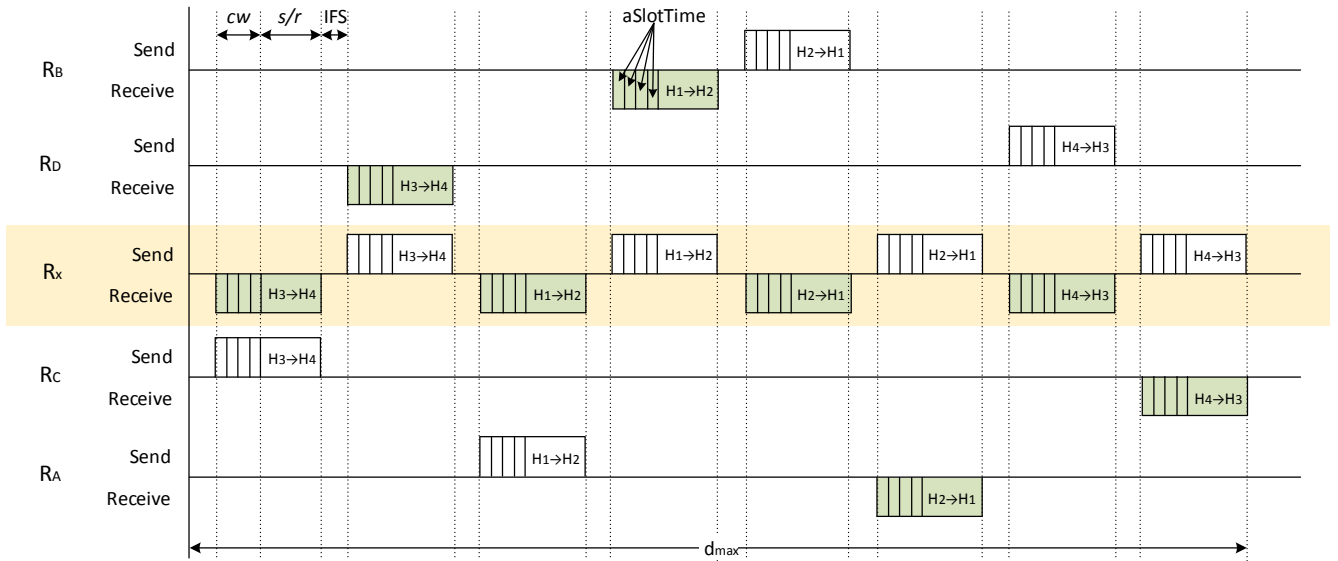


Figure 3: The timing diagram at router R_x according to Figure 2. Router R_x forwards packets between the pairs $R_A \leftrightarrow R_B$, $R_C \leftrightarrow R_D$. As can be seen here, the speed at which router R_x can forward all packets between its neighbors determines the overall delay d_{\max} of the process.

Table I: The resulting EDCA settings for the Tactile Coordination Function. In bold font the settings for the Tactile Control Access Category (AC_TC). All other settings remain the same as in IEEE 802.11 [3].

User Priority	Access Category	IFS	CWmin	CWmax
1, 2	AC_BK	7 slots	15	1023
0, 3	AC_BE	3 slots	15	1023
4, 5	AC_VI	2 slots	7	15
6, 7	AC_VO	2 slots	3	7
8	AC_TC	1 slot	3	7

and all packets have the same size, the maximum duration of this process is known. It depends on the worst-case time that is needed to transmit one packet, which is

$$d_{\max}^{\text{pkt}} = \text{IFS} + cw_{\max} \cdot \text{aSlotTime} + s/r.$$

s here denotes the length of a packet in bits and r is the bit rate of the channel.

Since each adjacent router transmits a packet and receives one, the total time for the transmission process at router R_x is $d_{\max} = 4 \cdot n \cdot d_{\max}^{\text{pkt}}$, assuming that n tactile flows intersect at R_x .

V. ANALYSIS OF QUEUEING DELAY

For multiple tactile application networks, we see that the capacity of the intersection router R_x is the most critical constraint. In this section, we analyze the stochastic behavior of this additional delay. For practical reasons, the calculation of worst-case delays lead to an over-estimation of the delay and therefore to an under-utilization of the network. We show that the average queueing delay can be expected to be very small for highly utilized systems and is even negligible in case of small utilization.

As was depicted in the previous section, the interference has to be resolved by scheduling. The packets that arrive at R_x and that cannot be forwarded immediately by R_x have

to be queued until they are in charge to be sent by the scheduler. We therefore model the additional delay caused at R_x as a queuing delay of a M/M/1-queue in Kendall's notation. We assume a poisson distribution in both the arrival and service processes. Both processes cannot be assumed to be deterministic because the medium access in IEEE 802.11, for example, can not provide any timeliness and is subject to transmission jitter (we could although assume determinism in a TDMA-based network.)

The average delay d_{avg} of an M/M/1-queue can be computed by

$$d_{\text{avg}} = \frac{\rho}{\mu - \lambda}$$

with arrival rate λ and service rate μ . $\rho = \lambda/\mu$ denotes the utilization. The delay is independent of the prioritization of the queue, and thus invariant regarding the scheduling strategy.

For our analysis, we assume a packet-oriented queueing system with a given packet length s . We further assume that $s=100$ Bytes, which is big enough to store a 6-degree-of-freedom vector of floats plus all overhead with flow meta information and packet headers. The waiting time of the first packet in line equals the time needed for the transceiver to transmit 100 Bytes over the medium. The system's service rate μ therefore depends on the medium's bit rate r and packet size s , or, more precisely, $\mu = r/s$. The packet arrival rate λ , on the other hand, is given by the application's QoS demands and the number of flows present in the system. The overall arrival rate at a router R depends on the number of flows n that are routed along this particular router: $\lambda = 4 \cdot n \cdot \lambda_{\text{Flow}}$. Factor 4 results from the two directions of each individual flow, multiplied by two to represent one transmission and one reception. We assume all flows to be equal in packet rate and packet size.

For our analysis, we choose three different queuing systems Q1–Q3 as shown in Table II. Q3 matches the constraints that are currently discussed in terms of the Tactile Internet idea: With the packet rate of 1000 Hz, applications like NCS, transparent tactile feedback and haptic communication will be possible. We however scale the bit rate of the channel accordingly with the applications. All three systems have the same per-flow utilization.

Table II: Three different assumptions for M/M/1 queuing systems. Service rates result from the respective transmission size divided by the packet size of 100 Bytes.

	Q1	Q2	Q3
Arrival rate (per flow) $\lambda_{\text{Flow}} [s^{-1}]$	10	100	1000
Bit rate r [Mbit/s]	0.5	5	50
Packet size s [bit]	800	800	800
Service rate $\mu [s^{-1}]$	625	6250	62500

Figure 4 shows the graph for d_{avg} as a function of n . The formula directly results from the above description.

$$d_{\text{avg}}(n) = \frac{4sn\lambda_{\text{Flow}}}{r^2 - 4rn\lambda_{\text{Flow}}}$$

The maximum utilization is reached at $n = 15.625$ in all three systems. We want to highlight three observations:

- Q2 and Q3 remain largely below the 1 ms mark: Q2 for $n \leq 13$ and Q3 for $n \leq 15$. The 1 ms mark denotes the most important QoS-constraint of tactile data flows. Q3 also represents the 1 kHz property of tactile data flows, while its bit rate is moderate: Present wireless transmission technology achieves bit rates that are higher by one or two orders of magnitude.
- The functions show steeper increases when n reaches the maximum capacity. For small n , however, the delay is approximated by $\hat{d}_{\text{avg}}(n) = k_2 \cdot n \geq d_{\text{avg}}(n)$. Using this linear function, and assuming k_2 is a global constant, the local delay model results in the global model of equation (1).
- As bit rate increases in one order of magnitude, the average delay decreases in one order, even despite the fact that the packet rate increases as well. So, with advancing transmission technology, we can expect a reduction in queuing delay.

VI. CONCLUDING REMARKS

In this paper, we have discussed the key challenges in communication of tactile applications within Wireless Multi-Hop Networks (WMHNs). The Tactile Internet idea is of big interest today as it enables many applications like Networked Control Systems, Teleoperation, Telesurgery, Haptic Communication and so on. As our everyday life becomes more and more mobile, we need to design these applications with wireless communication technologies in mind. WMHN will be an essential part of wireless networks in future, especially as part of cellular networks like 5G. The D2D functionality of 5G enables far better coverage and enables more applications in Pervasive Computing as ever before.

We depicted the two main aspects to control for meeting tight latency requirements of 1 ms, which are the number

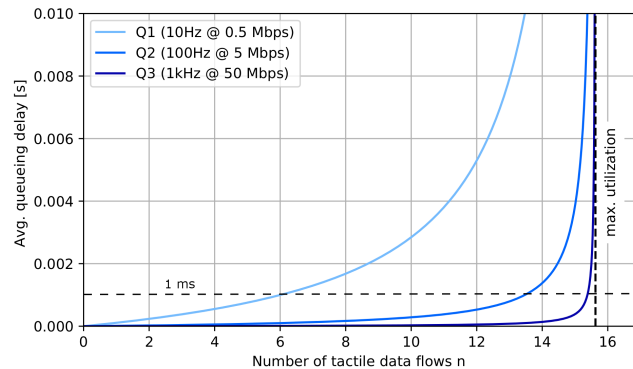


Figure 4: Queuing delay $d_{\text{avg}}(n)$ at router R_x in relation to the number of flows n for the different system assumptions Q1–Q3.

of intersecting flows n and the path length h . We also derived a formula for the queuing delay $d_{\text{avg}}(n)$ at a given router and motivated the use of a linear approximation. The resulting linear approximation model in equation (1) is able to determine delay bounds in WMHN without specific dynamic knowledge and may be promising to further develop capacity and delay models for WMHNs. We will focus on online calibration approaches for the parameters k_1, k_2 in future work, as well as on the constraints of the linear model.

As our theoretical analysis shows, the 1 ms-bound is realistic in IEEE 802.11 networks. Average delays can be expected to be very low, which is sufficient for a vast number of applications that do not require strong jitter bounds. The benefits of IEEE 802.11 over hard real-time, TDMA-based networks are high scalability, dynamic self-configuration, flexibility and better self-X properties. We believe that applications which can tolerate jitter and only need soft real-time environments benefit strongly from these properties.

We plan to evaluate applications in our own testbed for the Internet of Things, the Magdeburg IoT-Testbed [7], that consists of 60 nodes with IEEE 802.11, IEEE 15.04 and sub-GHz technology. The testbed will be extended to other technologies and bigger size in future, enabling a diverse analysis of Tactile Internet applications and Haptic Communication.

REFERENCES

- [1] M. Simsek, A. Aijaz, M. Dohler, J. Sachs, and G. Fettweis. 5g-enabled tactile internet. *IEEE Journal on Selected Areas in Communications*, 34(3):460–473, 2016.
- [2] A. Aijaz, M. Dohler, A. H. Aghvami, V. Friderikos, and M. Frodigh. Realizing the tactile internet: Haptic communications over next generation 5g cellular networks. *IEEE Wireless Communications*, 24(2):82–89, April 2017.
- [3] IEEE Std 802.11-2016: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Standard*, Dec 2016.
- [4] T. B. Sheridan and W. L. Verplank. Human and computer control of undersea teleoperators. Technical report, DTIC Document, 1978.
- [5] E. Steinbach, S. Hirche, M. Ernst, F. Brandi, R. Chaudhari, J. Kammerl, and I. Vittorias. Haptic communications. *Proceedings of the IEEE*, 100(4):937–956, 2012.
- [6] A. Nasrallah, A. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. ElBakoury. Ultra-Low Latency (ULL) Networks: A Comprehensive Survey Covering the IEEE TSN Standard and Related ULL Research. *arXiv preprint arXiv:1803.07673*, 2018.
- [7] MIoT Testbed. http://comsys.ovgu.de/MIOT_Lab.html.

Packet Merging-driven Tree Construction in Wireless Sensor Networks

Aleksandar Ilić

IHP

Frankfurt (Oder), Germany

ilic@ihp-microelectronics.com

Mario Schölzel

IHP

Frankfurt (Oder), Germany

schoelzel@ihp-microelectronics.com

Abstract—Performance of Medium Access Control (MAC) protocols used in convergecast sensor networks such as TreeMAC depends on topology of the data gathering tree used for routing data packets. We show that data rate can be increased if this topology is optimized during the tree construction phase. The algorithm that performs tree construction and optimization in parallel is proposed and then compared with two similar state of the art protocols.

Index Terms—wireless sensor networks, convergecast, data collection tree, minimum path tree, bipartite graph maximum matching

I. INTRODUCTION

One of the most common data collection scenarios in wireless sensor networks (WSN) is convergecast. It is collection of data from many sensors to one node which is called the sink. Depending on network application, the optimization goal for the data collection protocol in such networks is either minimization of the energy consumption, or maximization of the data rate. Data rate depends the most one the MAC (Medium Access Control) protocol. In the case of a sensor network in which traffic is generated by all nodes in the network and the data generation frequency is high, best results are obtained when a TDMA (Time Division Multiple Access) protocol is used. These protocols avoid collisions by allocating conflicted nodes in separate time slots. There are many algorithms that can optimize this allocation in terms of the data rate. However, before a schedule can be calculated, the network must be discovered, and the gathering tree formed, by allocating children groups to parent nodes. Apart from the TDMA schedule, data rate of such a network is also dependent from the data gathering tree structure. This paper proposes tree formation algorithm that helps increase the data rate by allowing the nodes to merge packets and transmit them together in one time slot.

In many-to-many networks, the optimal TDMA schedule is the shortest one i.e. the one that allows to all the nodes to transmit in the lowest number of slots while preventing conflicted nodes from transmitting at the same time. This schedule can be calculated using graph coloring algorithms. Such a schedule will be optimal only when all nodes have equal load (equal channel demand). However this is not the case in a convergecast network. In convergecast networks, nodes which are the closest to the sink have the highest load

(since they need to pass all the data from their children) and therefore require more time slots than the nodes at higher levels. The problem of finding the optimal schedule in this case is shown to be NP-complete by Choi *et al.* [1]. Most efficient protocols divide the gathering tree into top subtrees and then schedule the individual subtrees. A top subtree is a subtree rooted at one of the sink's children. One such protocol is TreeMAC, it achieves a schedule length equal to $3N$ where N is the number of nodes in the network excluding the sink [2]. Similar schedule length is reported in [3], while Grandham *et al.* [4] reports the same schedule length in general case, with the possible reduction when top subtrees are independent (can transmit at the same time without conflicts). This result has a theoretical value, but it will rarely be applicable in a real world scenario. Since most of the protocols for TDMA scheduling in data gathering trees are similar and produce the schedule of similar length, the proposed optimization is going to be demonstrated using the example of TreeMAC. It can however be applied to other similar protocols without modifications.

Packet merging is a technique used to increase network throughput by merging multiple smaller packets that have the same destination address [5]. It is possible to merge packets and still transmit them in one time slot because due to the technical reasons the time slot size will often be longer than the transmission time of an average packet. This technique is especially useful in convergecast networks because most of the packets have the same destination address, the sink. When applied to TreeMAC, this technique can reduce the schedule up to three times, depending on the tree topology. TreeMAC operates on shortest path trees and uses two-hoop interference model to schedule parallel transmissions. To allocate slots proportionally to the workload, it divides time into frames (one frame equals three time slots). The sink will be active all the time, while different branches of the tree are allocated in different time frames. If a branch contains n nodes, in general case $3n$ slots are required for scheduling this branch, and the sink receives one packet in each frame. When packet merging is used, the sink can receive up to three packets in one frame, providing that the topology allowed for the three packets to be merged.

The schedule formation and the topology dependence is going to be demonstrated using the two data gathering trees constructed on the same network shown in Fig. 1(a). In the

figure, circles represent sensor nodes, and their IDs are written inside the circles. A solid line represents the connection between a parent and its child, while a dashed line connects a parent with its possible child. It is assumed that all packets have length that allows for three packets to be merged and transmitted together in one time slot. The generated time schedule for the unoptimized network is shown in Fig. 1(b). In the first frame TreeMAC schedules nodes 1 and 7 to transmit in parallel in slot 3 of the frame. During this frame, packets from nodes 5, 3 and 1 are merged and transmitted to the sink. Meanwhile, the packet from node 7 is forwarded to node 5 and will reach the sink in the next frame. The rest of the schedule is formed in the similar manner. Note that in the third frame, only the packet from node 6 reaches the sink, because all nodes on the way to the sink have already transmitted a packet. If node 6 was assigned as a child to node 4 its packet would have been merged with those of nodes 2 and 4 and delivered to the sink together, reducing the schedule from 5 to 4 frames as shown in the Fig. 1(c). Therefore we deduce that in order to maximize packet merging, the number of leaf nodes the with height (distance from the sink) smaller than the maximal height of the tree needs to be minimized.

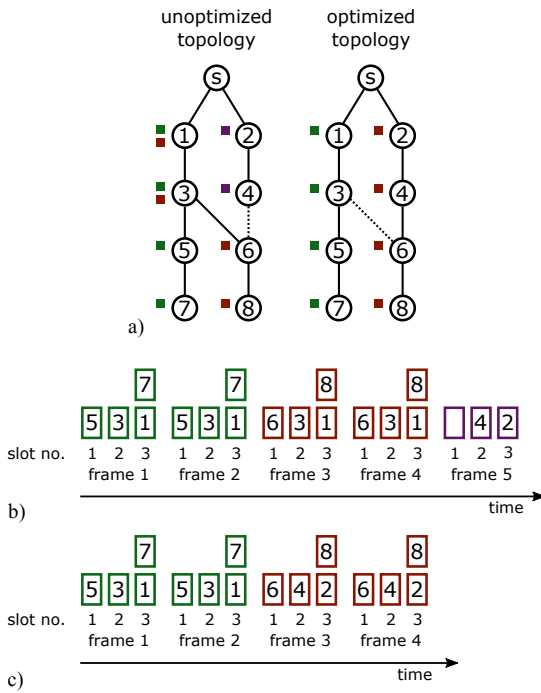


Fig. 1. TreeMAC schedule example

Section II presents an algorithm that performs optimization according to those constraints. The main difference between the proposed algorithm and the existing topology control techniques is in the definition of the optimal topology. The existing work focuses predominantly on load-balanced shortest path trees. Thereby was load-balanced defined in different ways. Hsiao *et al.* introduces the definitions of a fully load-balanced tree and a top load-balanced tree [6]. To evaluate the

result, weight of a tree is introduced as the total number of nodes in the tree. Top load-balanced tree is a tree in which the weight of each top subtree is equal, while the second one is the one in which all the branches have the same weight. Another definition, given by Andreou *et al.* defines a load-balanced tree as a tree in which the branching factor of each node does not exceed a certain value [7]. Load-Balanced trees were so far used to achieve different optimizations, such as increasing network lifetime [8], reducing number of collisions [7], or increasing network throughput [9]. Another approach is network optimization based on link quality [10]. However this was so far done for protocols that do not utilize parallel scheduling. Since networks with parallel scheduling are our focus, and implementation of such methods in these networks is connected with many difficulties, we have chosen to deal with link quality issue by simply avoiding usage of bad links when possible. To achieve this link quality is ascertained in the tree construction phase, whenever a new node is discovered. Another similar topic in topology control is minimum energy routing [11]. Those algorithms are however not applicable here because they would route the packets between nodes on the same level, which would introduce the need for a different parallel scheduling algorithm. In Section III we show that for the intended application, the proposed method performs better than the existing ones.

II. THE PROPOSED TREE CONSTRUCTION PROTOCOL

The goal of the algorithm is to construct shortest path spanning tree while minimizing the number of leaf nodes on the levels lower than the tree height. Since the tree must be a shortest path tree, children parent assignment is done between the nodes on two consecutive levels (level is the set of nodes at the same distance from the sink). The lower level is called the parent level, and the upper the children level. Nodes from the first level are called parents and nodes from the second children. A node on the parent level that does not get a child assigned becomes leaf node. Therefore, the optimal assignment is the one that minimizes the number of nodes on the parent level without children. Fig. 2 illustrates that this problem is not trivial even when the number of nodes on the both levels is the same. In the figure, if node $C2$ is assigned to node $P1$, node $P3$ will be left without children, even though it has two possible children Fig. 2 (b).

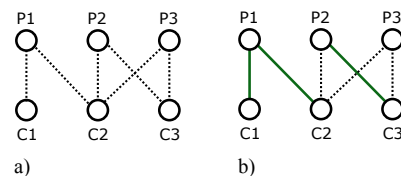


Fig. 2. Network example that demonstrates the assignment problem

A graph comprised of a parent level, a children level and the possible connections between the two levels is a bipartite graph (Fig. 2). The assignment problem can be divided into two steps. The first step is connecting maximum number of

children parent pairs in such a way that each parent can have maximum of one child, and each child maximum of one parent. If there are more children than parent nodes, some of the children will remain unconnected. Such children will have the possibility to connect only with parents which already have a child assigned. Thus, their assignment will have no effect on the number of nodes without children on the parent level, and they can be assigned to a parent in the second step using either random or link-quality based criteria. The first step represents problem of finding a maximum matching of a bipartite graph. Therefore in order to calculate children allocation, we first find the maximum matching using Ford-Fulkerson algorithm. Then we simply connect the remaining unconnected children to a possible parent with the best link quality or the smallest ID in case that link quality was not ascertained.

The complete network discovery and tree construction protocol is coordinated by the sink. Since parent assignment is calculated based on two tree levels and connectivity between them, the discovery and assignment is performed level by level. Algorithm executed by the sink to perform tree construction is shown in Fig. 3. In line 2 the sink discovers the first level. This is done by sending a *Query* message with the MAC address set to broadcast. Nodes that are not assigned yet will respond to this message and be discovered. After a node was discovered, a link quality estimation can be performed using an adequate method. If the link quality is not satisfactory the node will be removed from the list of discovered nodes. In line 6 nodes on the first level are assigned to the sink. This is done by sending a message to each of the nodes. Then the second level is discovered using the function *DiscNext()* in line 7. This function takes the list of nodes on the last assigned level as an argument, and instructs every node in this list to perform discovery of the next level. The function collects the responses, forms a list of nodes on the next level and their connectivity with the previous level, and returns this list. In line 9 the parent assignment for this newly discovered level is calculated using the previously described algorithm. Nodes on this level are then assigned to parents and subsequently asked to perform discovery of the next level (lines 10-12). This is repeated until no nodes are discovered on a certain level (while loop starting at line 8). Because in i^{th} step, the nodes at the height i are discovered and then assigned to a node on the previous level, it is obvious that the formed tree will be a shortest path tree.

III. EVALUATION

To evaluate the proposed algorithm, we are going to compare it with the state of the art top-load balancing algorithm proposed by Incel *et al.* [12] and the balancing algorithm proposed by Andreou *et al.* [7], called ETC (Energy Driven Tree Construction). The first one utilizes search sets to determine the optimal parent for a child. A search set identifies nodes on up to two levels above the child that will be left with only one possible branch to connect in the future if the child is allocated to a certain parent. A child chooses the parent for which the sum of its subtree weight plus the search set size

```

1: procedure CONSTRUCTTREE
2:    $parentLevel \leftarrow DiscLevel1()$ 
3:   if  $parentLevel.count = 0$  then
4:     return
5:   end if
6:    $AssignLevel1(parentLevel)$ 
7:    $childrenLevel \leftarrow DiscNext(parentLevel)$ 
8:   while  $childrenLevel.count > 0$  do
9:      $asg \leftarrow Assing(parentLevel, childrenLevel)$ 
10:     $Assign(childrenLevel, asg)$ 
11:     $parentLevel \leftarrow childrenLevel$ 
12:     $childrenLevel \leftarrow DiscNext(parentLevel)$ 
13:   end while
14: end procedure

```

Fig. 3. Tree construction algorithm

is minimal. ETC algorithm defines maximal branching factor as $\sqrt[h]{N}$, where h is the tree height and N is the number of nodes in the network. If this factor is exceeded at a certain node, it will ask some of its children to try to find another parent. This will not always be successful, and therefore the tree produced is called near-balanced tree.

The comparison is performed on the example network that is shown in Fig. 4(a). We are going to demonstrate the drawbacks of the existing methods using this network and show how the proposed method overcomes them. The more extensive comparison is going to be presented at the presentation in the form of simulation results. The data gathering tree produced by the top load balancing algorithm is shown in Fig. 4(b). We can see that in this case, no children will be assigned to node 2, even though node 4 was available as its possible child. This happened because when child 4 was assigned to a parent, both possible parents, node 1 and node 2, were in branches with the same weight (equal to one). Second criteria used, search set, was also the same for both parents. Both search sets in this case contained only node 7, because this is the only node that will have to join the same branch as node 4 (node 8 will be able to choose between subtrees rooted at node 1 and at node 3). Therefore the tie was broken on the smaller index, and node 1 became the parent of node 4. We also observe that for node 5, the algorithm worked as expected. In that case, the search set for node 1 contains nodes 8 and 9, and for node 3 it contains only node 9. Therefore node 5 was assigned to node 3, allowing for the nodes 8 and 9 to be shared between two subtrees in the final step.

The network constructed by the ETC algorithm is shown in Fig. 4(c). Since the First-Heard-From principle is used to create the tree and node 1 was the first to perform assignment, both node 4 and node 5 were assigned to it, leaving node 2 without children. Because branching factor for this network is $\beta = \sqrt[3]{10} = 2.15$, node 1 will not try to re-assign any of its children. Finally, Fig. 3(d) shows the network obtained using the proposed algorithm. As it can be seen, the maximum matching for the bipartite graph constructed from levels 1 and

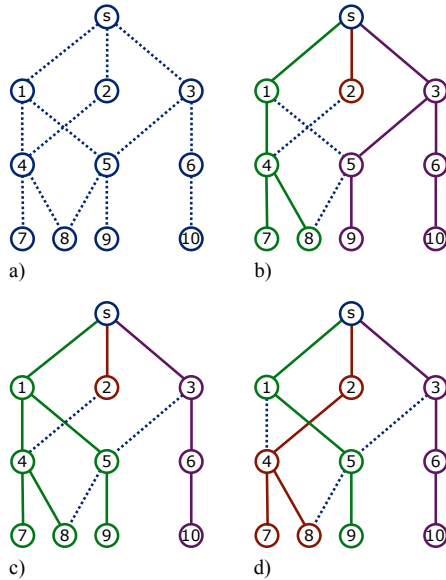


Fig. 4. Comparison of different algorithms (a) the network, (b) top load balancing, (c) ETC, (d) the proposed algorithm

2 contains three edges. The algorithm has found this matching and used it to perform child assignment. The constructed tree is optimal for packet merging and allows scheduling the network in 4 time frames using TreeMAC. For scheduling the other two networks, 5 time frames would be required. We also note that the resulting tree in this case is top load-balanced since the maximum difference between weights of the subtrees is equal to one, even though this was not the intention of the algorithm. This leads to a conclusion that the algorithm could be modified to construct top load-balanced trees, which is left as a topic for future work.

IV. CONCLUSION

In this paper, the data gathering tree construction protocol for creating shortest path trees is proposed. The protocol is aimed to be used with TreeMAC or similar TDMA MAC protocols that utilize parallel node scheduling and packet

merging to increase throughput. We show that in these protocols scheduling reduction using packet merging is dependent on the network topology. We then introduce new criterion for network optimization, the number of leaf nodes. The tree construction algorithm that constructs a shortest path tree is then introduced. To optimize the tree topology, the parent assignment needs to be calculated to qualify the defined criterion. The parent assignment algorithm that finds the optimal assignment in polynomial time if it exists is therefore proposed. The proposed algorithm is compared with two state of the art algorithms. The comparison shows that in the considered application scenario, the proposed algorithm performs better.

REFERENCES

- [1] H. Choi, J. Wang and E. Hughes, "Scheduling for information gathering on sensor network," *Wireless Netw.*, 2007.
- [2] W. Song, H. Renjie, B. Shirazi, R. LaHusen, "TreeMAC: Localized TDMA MAC protocol for real-time high-data-rate sensor networks," *Pervasive Mob. Comput.*, vol. 5, nr. 6, 2009.
- [3] S. Ergen, P. Varaiya, "TDMA scheduling algorithms for wireless sensor networks," *Wireless Netw.*, vol. 16, pp. 985-997, 2010.
- [4] S. Gandham, Y. Zhang and Q. Huang, "Distributed time-optimal scheduling for convergecast in wireless sensor networks," *Computer Networks*, vol. 52, nr. 3, pp. 610-629, 2008.
- [5] V. Akila, T. Sheela and G. A. Macrigna, "Efficient Packet Scheduling Technique for Data Merging in Wireless Sensor Networks," *Chinacom*, vol. 14, nr. 4, pp. 35-46, April 2017
- [6] P. Hsiao, A. Hwang, H. T. Kung, and D. Vlah, "Load balanced routing for wireless access networks," *Proc. IEEE Infocom*, pp. 986-995, 2001
- [7] P. Andreou, A. Pamboris, D. Zeinalipour-Yazti, P.K. Crysanthos and G. Samaras "ETC: Energy-drive Tree Scheduling in Wireless sensor networks, Tenth International Conference on Mobile Data Management: Systems, Services and Middleware, 2009.
- [8] J. He, S. Ji, Y. Pan and Y. Li, "Constructing load-balanced data aggregation trees in probabilistic wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1681-1690, July 2014.
- [9] H. Zhang, F. Österlind, P. Soldati, T. Voigt and M. Johansson, "Rapid convergecast on commodity hardware," *SECON*, 2010.
- [10] W.B. Pöttner, H. Seidel, J. Brown, U. Roedig and L. Wolf, "Constructing schedules for time-critical data delivery in wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 44, April 2014.
- [11] P. Santi, "Topology control in wireless ad hoc and sensor networks," Wiley, 2005.
- [12] Ö. D. Incel, A. Ghosh, B. Krishnamachari, and K. Chintalapudi, "Fast data collection in tree-based wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 11, nr. 1 pp. 86-91, January 2012.

Managing Swarms of Robots: From Centralized Scheduling to Decentralized Scheduling

(work in progress)

Daniel Graff

Kommunikations- und Betriebssysteme
Technische Universität Berlin
 Berlin, Germany
 daniel.graff@tu-berlin.de

Reinhardt Karnapke

Verteilte Systeme / Betriebssysteme
Brandenburgische Technische Universität Cottbus–Senftenberg
 Cottbus, Germany
 karnapke@b-tu.de

Abstract—Programming swarms of robots individually is tedious and error prone. Therefore, we introduced the Swarmscheduler in previous work. It takes care of the assignment of tasks to individual robots, based on a simple 4-tuple task description supplied by the user. However, when the swarm becomes really large, containing thousands of robots, the Swarmscheduler will become a bottleneck due to its centralized architecture. Therefore, we are working on decentralizing the Swarmscheduler. In this paper we present two different approaches for decentralization we are currently investigating, and present some preliminary results.

Index Terms—scheduling, swarms, robots, decentralized management

I. INTRODUCTION

Future Cyber-physical Systems might be comprised of a lot of individual nodes that should work together in order to fulfill a user specified task. Whether this involves chores like cleaning the floor in an automated home, welding a workpiece in an automated factory or tracking and following a group of sharks in the ocean does not matter. The important fact is that multiple embedded systems need to cooperate in order to accomplish something. When the number of mobile embedded systems (robots) is increased drastically, we get to the notion of *the Swarm*. The swarm can be used by a group of users and their applications concurrently. However, programming a large number of nodes individually is tedious and error prone. Therefore, suitable abstractions are needed.

In previous work we introduced the Swarmscheduler [1]–[3], which provides these abstractions. Users specify a task with four properties: Required sensors, required actuators, a time, and a place. This information is given to the Swarmscheduler, which then first identifies which of the existing mobile robots under its control have the required sensors and actuators. From this group, all robots that already have a task to fulfill at the requested time are deducted. For the remaining robots, a check is performed, whether they could reach the requested place in time without violating any of their previous assignments. Also, path and velocity planning is performed, taking stationary and mobile obstacles into account, as this might delay the robot further. The algorithm then delivers a

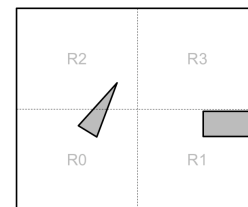


Fig. 1. Splitting the world into static regions. Grey shapes represent obstacles

number of candidate robots, from which the most suitable one is chosen and informed that it has a new task to fulfill.

Even though this system already works quite well, there are a number of possible improvements. One of these, which will be addressed in this paper, is scalability. The current version of the Swarmscheduler is a centralized one, which of course makes planning much easier as it has complete knowledge of the world. However, centralized approaches do not scale well, which will become a problem for the envisioned size of thousands of nodes in the swarm. To tackle scalability, decentralized and hierarchical approaches can be investigated. In this paper we describe the ongoing work of transforming the existing Swarmscheduler into a decentralized one, by investigating two different options. Option one is a static partitioning of the world with the resulting challenges when nodes cross boundaries, while option two is a dynamic approach where nodes form ad-hoc scheduling groups which need to be constructed according to certain criteria.

The remainder of this paper is structured as follows: Section II describes the static approach in more detail, followed by the dynamic approach in section III. Some preliminary results are given in section IV. We finish with a conclusion and future work in section V.

II. THE STATIC APPROACH

In the static approach, the world (or at least the area in which our robots are active) is split into different regions, with a separate scheduler for each region (figure 1). However,

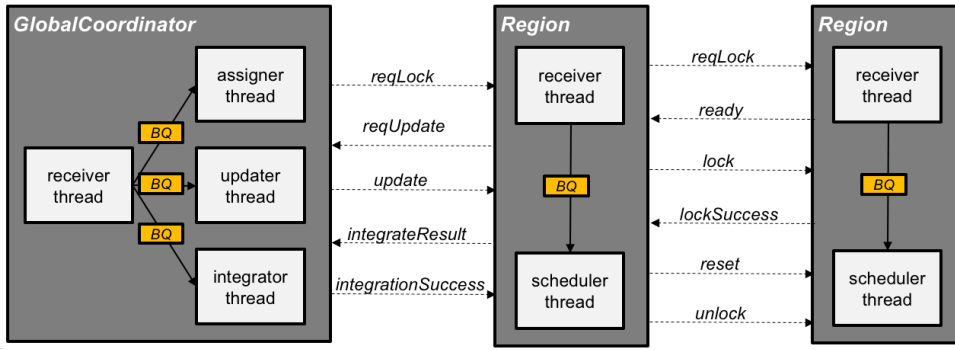


Fig. 2. Flow chart of the scheduling of a task that involves two regions

there is still a need for a central coordinator, which represents the interface between swarm and user. When a user has a new job to schedule, he/she transmits the task description to the central coordinator. The coordinator checks the location of the requested task and forwards the task description to the scheduler that manages the region in which this task should be fulfilled. Please note that this way the user interface that has been used by the centralized approach can be kept, the user does not need to know about the different regions or that a decentralized approach is used at all.

When a regional scheduler receives a task description it tries to schedule the job locally. When it has succeeded, it informs the node(s) that should take care of the task. However, there might be problems when tasks require nodes to move across region boundaries. In order to prevent collisions and to enable schedulers that do not have enough nodes in their region to fulfill a task to request nodes from their neighbors, a global schedule is also maintained. This is done by the global coordinator. Every time a local scheduler has made a decision, it also informs the global coordinator of the new schedule it has calculated. The global coordinator then tries to integrate the new local schedule into the *single* global schedule, and informs the local scheduler of the outcome. If the global coordinator finds problems with the new local schedule, the regional scheduler needs to revoke its plan and start over with a new candidate robot.

Figure 2 shows the flow chart for the scheduling of a job that requires the interaction of two regional schedulers. First, the received task description is transmitted from the global coordinator to the region where the task shall be scheduled, requesting a lock on this region. The regional scheduler realizes it needs to cooperate with another region and requests the lock on the region on the right, which is answered with a ready message. The first region then requests an update from the global coordinator and locks the second region, which replies with a lock success. Please note that this two step locking approach is necessary in order to avoid deadlocks when multiple regions try to lock the same neighboring region. Now that the first region holds all the resources it needs, it requests the global coordinator to integrate the new local schedule into the global schedule. When it receives the

integration success (or failure) message, it transmits an unlock (or reset) message to the locked second region, so that the second region is once more free to schedule something itself or be acquired by another region which needs its support.

III. THE DYNAMIC APPROACH A.K.A. DYNAMIC SCHEDULING GROUPS

The drawback of the static approach is that there is a fixed amount and size of regions during system execution. In the worst case, if all schedule requests point to locations in the same region, then we have the same situation as in the centralized case and no distribution of workload. As a result the performance degenerates. Configuring the amount of regions as well as their size perfectly requires further domain knowledge about job distribution. In order to overcome those issues, we present the dynamic approach.

In the dynamic approach regions are created on demand as a result of a schedule request. Those regions are also called scheduler groups. Each group handles exactly one request and is created at the location where the job shall be executed. Subsequently, nodes are added that are in certain proximity. Groups are created temporarily and disappear after a job has been scheduled. So, they usually only exist for a short amount of time. The nodes locations associated with a certain group span the region. They may differ in space and time. There is no limitation in number of regions as they are automatically garbage collected when they “expire”. The expectation is to get even better performance and be scalable independent of a certain configuration. Thus, no domain knowledge is required. The central coordinator as introduced in Chapter II is used here as well with slight modification.

Let us start with an example. Figure 3(a) shows an initial situation. The world contains two static obstacles (large rectangles). The blue diamonds represent jobs and the yellow triangles represent nodes. Jobs appear in the order $\diamond_1, \diamond_2, \diamond_3, \diamond_4$. The first scheduling request creates a scheduler group with nodes \triangle_1 and \triangle_3 as depicted in Figure 3(b). The corresponding region (r_1) is shown in Figure 4. The region is created by the union set of the bounding boxes of the ideal trajectories of \triangle_1 and \triangle_3 . Ideal means a node is able to reach the job location on straight lines. So, the path-planning part, therefore,

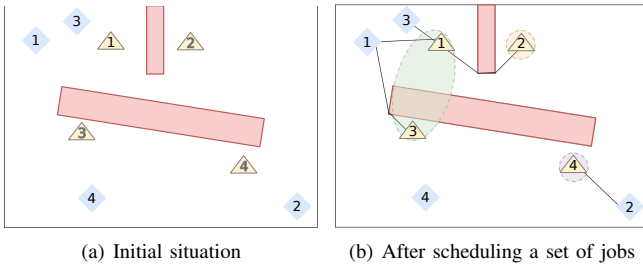


Fig. 3. The world and the emergence of scheduler groups

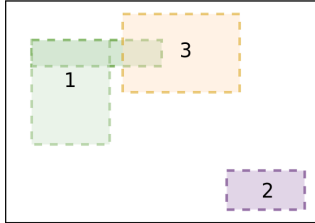


Fig. 4. Creation of region locks

avoids static obstacles, but does not take dynamic obstacles into account. The simple path-planning is computed very fast in comparison to the entire trajectory planning as it considers only static entities in 2D space. Time is neglected in this step as it is only taken into account when computing the entire trajectory.

Similar to the static approach a region is locked when performing the scheduling. The scheduling requests of \diamond_2 and \diamond_3 create r_2 and r_3 , respectively. All regions that have no overlap, i.e., the intersection set of regions is the empty set, are able to perform the scheduling within their groups in parallel. If the intersection set is not empty, then the scheduling request creating the latter region that leads to the overlap must be deferred until the other region lock is released.

Since there is no overlap in r_1 and r_2 , they can be processed in parallel. Region r_3 overlaps with r_1 , therefore, it is deferred until the scheduling request of \diamond_1 is finished. The scheduling of \diamond_4 is neglected here for clarity, but follows the same procedure. Similar to the static approach all data (global schedule) is kept at the global coordinator.

IV. PRELIMINARY RESULTS

For the simulation, we have set up a cluster of 25 Intel(R) Core(TM) i3-6100U CPU @ 2.30GHz processor with two cores. We simulated each node on a separate core. The simulation aimed to figure out in which configurations the centralized version performs better than the decentralized one (dynamic approach) and vice versa. As of now we only have results for the dynamic approach that we also call the decentralized approach. We are still working to get results from the static approach.

In each simulation run, we place the number of robots using a uniform distribution in the world. Spatio-temporal constraints, i.e., time and place, for the jobs are generated

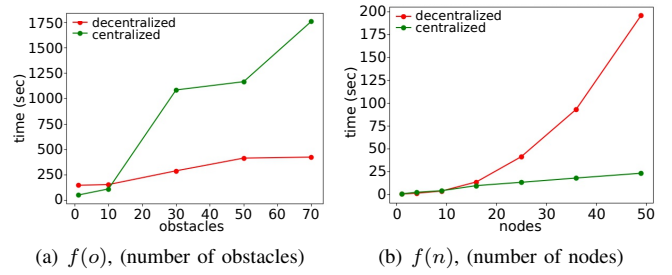
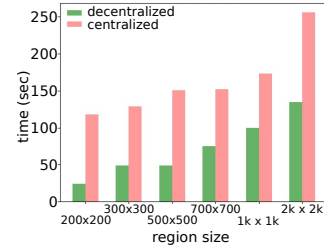


Fig. 5. Time: Centralized vs. decentralized

Fig. 6. $f(r)$, time as a function of region size

randomly: we randomly choose a 2D point in the world and create a 1×1 -square around it, then we randomly generate a time interval in which the job has to be executed.

Figure 5(a) shows the execution time of scheduling an amount of 10,000 jobs as a function of the number of static obstacles. The number of nodes is set to 25. Increasing the number of static obstacles results in a significant higher execution time in the centralized approach while the decentralized approach only shows a slight increase. From 10 obstacles onwards the computation time for the central scheduler is worse compared to the decentralized one. But why do static obstacles influence the computation time? Having more static obstacles results in a higher degree of path segments for a certain path in order to avoid collisions. When doing the trajectory computation each segment has to be checked for collision with dynamic obstacles. Although there is already a heuristic used which makes the computation faster, it still requires more time. Especially, when there is a collision it has to be resolved by adjusting velocity profiles. The higher computation time has a stronger impact on the centralized approach.

Figure 5(b) shows the execution time as a function of the number of nodes. The number of static obstacles is set to 20 and the number of jobs is set to 500. Using this combination shows a better performance of the centralized scheduler as those numbers are very low compared to the first case.

Figure 6 shows the execution time as a function of region sizes. The number of nodes is set to 25 and the number of jobs is set again to 10,000. Independent of the region size, the decentralized approach always outperforms the central scheduler.

Finally, we compared the job acceptance rate of both approaches as shown in Figure 7(a) and Figure 7(b). A job

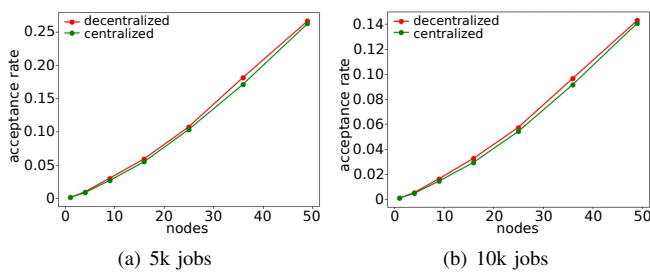


Fig. 7. Acceptance rate: Centralized vs. decentralized

acceptance rate of 1 indicates that all jobs have been scheduled successfully. However, in our simulation it was interesting to examine the acceptance rate when scheduling way more jobs than the nodes could actually accept. Please note that jobs have spatio-temporal constraints, i.e., they must be scheduled accordingly. Narrowing the time interval so much that only a fraction of them can be scheduled—as seen here in this example—shows the behavior of adding more nodes to the system which leads to a higher degree of linear increase in the acceptance rate. However, the curve behavior for the centralized and the decentralized approach is similar.

As of now we were only able to produce preliminary results and not a full evaluation. So, there might be additional aspects that we will find out when performing the full evaluation.

The decentralized approach is not designed and not intended to handle small setups efficiently. Especially latency has a bad impact on small setups. The current implementation sends the full schedule to other nodes which is as of now very time consuming. We expected a solid improvement when transmitting only the changes and thus reducing the message size significant.

V. CONCLUSION AND FUTURE WORK

Centralized control over a number of robots offers the advantage of full knowledge and, therefore, eases the computation of a global schedule which avoids conflicts and collisions. However, it suffers from scalability problems. In this paper we presented two different approaches we are currently investigating, a static partitioning of the world and the formation of dynamic, short lived scheduling groups. Both approaches have advantages and disadvantages, as evident in the preliminary results we have shown.

We plan to continue working on these two approaches, and to rank them against each other. The final outcome should be a set of settings, in which the static approach should be preferred and a set of settings, in which the dynamic scheduling groups perform better, so that a potential user can choose the right approach depending on the circumstances in which the swarm will be deployed. Also, as the user interface remains the same no matter which approach (centralized, static, or dynamic) is used, it should be possible to switch between scheduling approaches at runtime. This is another topic we would like to investigate.

REFERENCES

- [1] D. Graff and R. Karnapke, “Spatio-Temporal Coordination of Mobile Robot Swarms,” in *41th IEEE Conference on Local Computer Networks*. IEEE, November 2016. [Online]. Available: <http://ieeexplore.ieee.org/document/7796829/>
- [2] —, “The Swarm as a Service: Virtualization of Motion,” in *41th IEEE Conference on Local Computer Networks Workshops (LCN Workshops)*. IEEE, November 2016. [Online]. Available: <http://ieeexplore.ieee.org/document/7856147/>
- [3] D. Graff, “Programming and Managing Swarms of Mobile Robots: A Systemic Approach,” Ph.D. dissertation, Technical University of Berlin, Germany, 2017. [Online]. Available: <http://nbn-resolving.de/urn:nbn:de:101:1-201804164492>

Evaluation of the 6TiSCH network formation

Dario Fanucchi

Department of Computer Science
University of Augsburg
Augsburg, Germany

d.fanucchi@informatik.uni-augsburg.de

Barbara Staehle

Department of Computer Science
University of Applied Sciences Konstanz
Konstanz, Germany

barbara.staehle@htwg-konstanz.de

Rudi Knorr

Fraunhofer Institute for Embedded Systems
and Communication Technologies ESK
Munich, Germany

rudi.knorr@esk.fraunhofer.de

Abstract—This extended abstract focuses on the network formation procedure proposed by IETF 6TiSCH working group, a mandatory phase before nodes may transmit any sensed data. We evaluate through simulations the impact of TSCH- and RPL-parameters on the duration of and the charge consumed for the network formation process. We describe how to avoid an unsuccessful network formation and we give simulation-based recommendations for an appropriate parameter setting on typical network topologies.

I. INTRODUCTION

An open, standardised protocol stack proposed by IETF, denoted in the following as IIoT-stack, is nowadays emerging in industrial wireless communication [1]. The IIoT-Stack depicted in Fig. 1 is aiming to replace proprietary technology such as WirelessHART and ISA100. In this architecture, Time Slotted Channel Hopping (TSCH) at MAC layer [2] and the IPv6 Routing Protocol for Low power and Lossy Networks (RPL) [3] are the two standards with the most impact on guaranteeing a timely, reliable and energy efficient communication.

TSCH adopts time-slotted channel access and a frequency hopping technique. The time is organised as a continuously repeating sequence of slotframe formed by several timeslots, and up to 16 different physical frequencies are available for transmission at each timeslot. As a result, TSCH provides a matrix of links (or cells) for scheduling communications in the network. There are four link types: Tx, Rx, Shared and Idle. Tx and Rx links are dedicated links, allocated to a single sender-receiver couple for transmitting and receiving a data frame and its related acknowledgement (ACK). CSMA-CA regulates the transmission on shared links. Finally, nodes turn their radio off for saving energy on idle links.

RPL is a distance-vector routing protocol, which organises nodes as a Destination Oriented Directed Acyclic Graph (DODAG), a directed tree, rooted at the sink, which is usually responsible for data collection. Employing the concept Objective Function, RPL offers a flexible way to optimise the network topology, defining which metrics and how these are used to compute a node's rank, a gradient value which encodes the distance of each node from the sink.

The IETF 6TiSCH standardisation efforts and recommendations aim to face the problem of building and maintaining multi-hop schedules for RPL-organized, TSCH-based networks [4].

This extended abstract explores the so-called 6TiSCH *minimal configuration* (6TiSCH-MC) [5], a standard strategy proposed by IETF 6TiSCH working group for the network formation phase, where each node synchronises on a fundamental TSCH matrix and joins a DODAG. This initial phase is mandatory, before nodes may transmit any sensed data, and it is characterised by an interaction between TSCH and RPL protocols and their parameters. Answering the question *how long the 6TiSCH-network formation process will take?* is not trivial. Apparently, the size of the network has an impact. Moreover, using the IIoT-stack with its default parameters may lead to a very long time for (or even unsuccessful) network formation. We evaluate through simulations the delay and the consumed energy for this initial phase. Implementers of IIoT-solutions will benefit from our guidelines for an appropriate parameter setting, depending on some typical physical topologies.

Next, Section II briefly describes the way a 6TiSCH-network is formed and Section III presents the methodology and results of our simulative study. We draw some conclusions and gives an outlook on future works in Section IV. Our peer-reviewed conference paper [6] offers the interested reader an overview of related works and more details on outcomes.

II. 6TiSCH NETWORK FORMATION PROCEDURE

As hereafter described, the 6TiSCH-MC [5] primarily proposes (1) a strategy to allocate the transmission of control messages during the network formation and (2) coordination between the TSCH network-wide synchronisation and the RPL DODAG construction, which is crucial in this phase.

The 6TiSCH network formation starts with the sink (or root), which advertises the network presence by broadcasting Enhanced Beacon (EB) frames and DODAG Information Element (DIO) packets. EBs are generated every t_{eb} and contain all the necessary time information to allow the initial synchronisation among nodes, including the definition of the *minimal schedule*, i.e. a static communication schedule formed by only one shared link. DIO packets are ICMPv6 control messages, which announce the DODAGID, the sender's rank and other configuration parameters used for DODAG construction and maintenance. The Trickle algorithm controls the generation of DIO messages as specified in [7]. Specifically, the time is split into intervals of size I and the root triggers the generation of a DIO message at a random instant τ in the second half of each interval. The size I is varied over the time. In particular, the

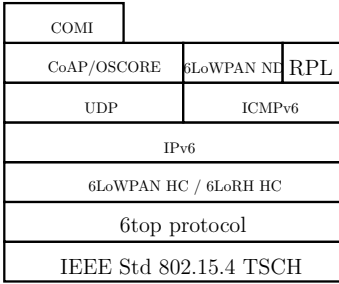


Fig. 1. IETF-Stack for IIoTs

root doubles I at the end of each interval, starting from the minimum size I_{min} , until a maximum value $I_{max} = 2^M \cdot I_{min}$ is reached. The root exploits the shared slots, offered every $N_s \cdot t_s$ by the *minimal schedule*, for the transmission of EBs and DIOs located in the outgoing queue. This process is exemplified in Fig. 2, where $t_{eb} = 200ms$, $I_{min} = 128ms$, and the *minimal schedule* are used. After being generated, EBs and DIOs are put in the transmit queue and consumed by TSCH in the next active slot. As illustrated, EBs are queued with a priority higher than DIOs.

When a node wishes to join the network, it uses preferably passive scanning, i.e., it turns the radio on to a randomly chosen frequency, and it listens for EBs. While waiting for a valid EB, the joining node keeps its radio always on and changes frequency every t_{scan} . After hearing a valid EB, a node learns the *minimal schedule* in the network, so it knows when to wake up for receiving or sending control frames related to the network formation procedure. The joining nodes will not start advertising the network presence straight away, but only after it has received a DIO message, computed its rank and selected its preferred parent among a set. This expedient assures a multi-hop time synchronisation with a loop-less structure since it reuses the DODAG structure with no extra signalling at MAC layer [8]. Optionally, a newly synchronised node can multicast a DODAG Informational Solicitation (DIS) to ask for a DIO and to speed up the joining.

This process spreads gradually to cover the whole network: each node may become an advertiser node after hearing an EB and after joining a DODAG. Each advertiser nodes will periodically send EBs. Since all control traffic listed above is transmitted within the single shared timeslot provided by the *minimal schedule*, collisions are likely to occur as more and more nodes join the network.

III. PERFORMANCE EVALUATION

A. Methodology

We evaluate the performance of the initial 6TiSCH-network formation using Cooja, the network simulator distributed as part of the Contiki-OS [9]. In the simulations, we use the Cooja mote type, i.e., a virtual hardware without limitations concerning memory and computation capabilities. As wireless propagation model, we use the Unit Disk Graph Medium

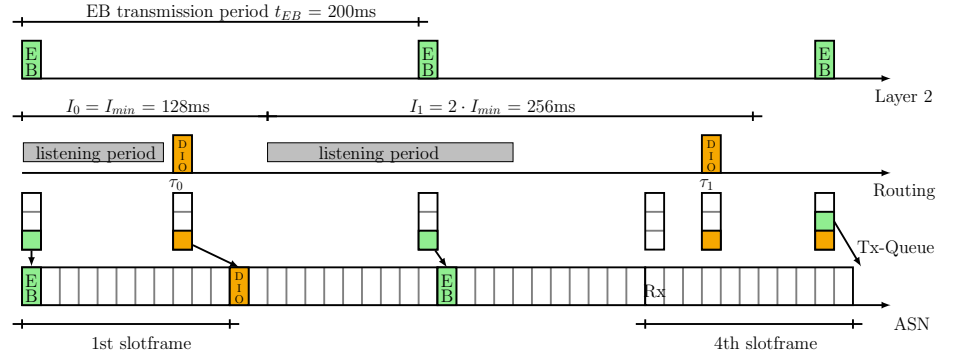


Fig. 2. A timeline of the network formation on an advertiser node

TABLE I
SETTING FOR TSCH- AND RPL-PARAMETERS

Parameter	Symbol	Value
TSCH slotframe length	N_s	101
TSCH timeslot duration	t_s	10 s
TSCH scan interval	t_{scan}	1 s
TSCH number of channel	N_c	4, 16
TSCH KA period	t_{ka}	{12, ..., 60} s
TSCH EB period	t_{eb}	{2048, 4096, 8192, 16384} ms
RPL redundancy constant	c	5
RPL interval doubling	M	8
RPL minimum interval	I_{min}	{128, 256, ..., 4096} ms

(UDG). Links are symmetric, the packet delivery ratio (PDR) is 100% in a transmission range of 50m, and the interference range is 100 m. We did not consider a realistic hardware and radio setting to narrow down the side effects and to study the protocol mechanisms alone.

We consider three categories of network topology and three different sizes $N_{size} \in \{9, 16, 25\}$:

- *Grid*: the sink is placed in the left-high corner and the distance between nodes is set, so that the degree of the central node is $d = 4$ with $N_{size} = 9$, or $d = 6$ with $N_{size} = 16$ and 25.
- *Ellipse*: for every node is degree $d = 2$, i.e., each node has a PDR of 100% only with the left and right one-hop neighbours, and the multi-hop path from a sensor node to the sink has a maximum depth $maxD = \lfloor N_{size} / 2 \rfloor$.
- *Random*: nodes are randomly placed in a square area of $100m^2$. The topology features a portion of the network with high density and other some nodes with a minimum degree $d = 2$.

For each simulated setting, i. e. a possible combination of parameter's value, we run 50 independent replications with different seeds, and we report the average value of each metric with its 95% confidence interval.

The configuration setting of TSCH and RPL are reported in Table I. Advertiser nodes generate EB frame with a random EB period in the value ranges between $t_{eb} \cdot [0.75, 1)$. Although the 6TiSCH-MC does not mention this mechanism, open-source implementations of the TSCH (e.g. OpenWSN and Contiki) include it, and [10] shows its advantages in the network synchronisation.

For the evaluation, we study the effects of TSCH EB period t_{eb} , TSCH number of channel N_c and RPL minimum interval I_{min} on the following performance metrics:

- *TSCH synchronisation time*, defined as the time between the transmission of the first EB by the PAN coordinator and the first network-wide TSCH synchronisation.
- *DODAG formation time*, defined as the time until all nodes are simultaneously in the DODAG.
- *Charge consumed* during the network formation, estimated summing the charge consumed by the radio component of each node (except the sink), when they are active or inactive.

B. Simulation of 6TiSCH minimal configuration

Table II reports on the number of simulations in which the network formation phase is completed within 30 minutes. As can be seen, an improper choice of t_{eb} and the number of channel offset N_c in relation to the network density (i.e. topology) can lead to an unsuccessful network formation, where at least one node is even after 30 minutes not yet operational and cannot transmit any sensed data to the sink. With only one shared slot for broadcasting control frame and $t_{eb} < 4 \cdot N_s \cdot t_s$, the positive effect of applying randomness to the EB periods can not take place, i.e. advertiser nodes choose the same slotframe for sending their EB, causing EB collision. The problem of EB collisions becomes more and more apparent as the network density increases (i.e., when passing from scenarios with $N_{size} = 9$ to them with $N_{size} = 25$). Compared to a setting with $N_c = 4$, we observed a linear increase of the network formation time, when $N_c = 16$ is used. For these reasons, we will consider only results obtained with $t_{eb} \geq 8192$ ms, $N_c = 4$ and $N_{size} = 9$ in the remainder of the section.

In Fig. 3 the solid line reports the *TSCH synchronisation time* and the dashed line depicts the *DODAG formation time*. These metrics are expressed as a function of the RPL minimum interval. As it can be seen, I_{min} shows an influence on the network formation time of random and grid topologies, which is worth investigating. The behaviour is pretty different from the results presented in [11] for IEEE 802.15.4-2011, where decreasing the RPL minimum interval yields to a reduction of DODAG formation time. One reason is that the configuration of the TSCH slotframe interferes with the Trickle mechanism and implicitly sets a range for the optimal value of I_{min} . As expected, when $I_{min} \ll N_s \cdot t_s$ the transmission of RPL packets, generated by the Trickle algorithm, may be significantly delayed or dropped in the transmission queues. We do not appreciate the influence of the RPL minimum interval when we deal with a sparse topology as the ellipse.

This fact can be explained considering the limited effect of the polite gossip policy (used by Trickle) on this type of network. Another insight from Fig. 3 is the significantly different elapsed time between the network-wide TSCH synchronisation and the DODAG completion, which we observe in the three topologies. The higher physical density in the random or grid topologies makes collision of control frames more likely to

TABLE II
SUCCESSFUL (%) DODAG FORMATIONS WITHIN 30 MIN

t_{eb}	N_c	Grid			Ellipse			Random		
		N_{size}			N_{size}			N_{size}		
2048	4	9	16	25	9	16	25	9	16	25
	16	0	0	0	70	16	0	0	0	0
4096	4	84	0	0	100	100	86	46	0	0
	16	12	0	0	100	100	26	2	0	0
8192	4	100	20	0	100	100	100	100	2	0
	16	96	0	0	100	98	40	66	0	0
16384	4	100	70	0	100	100	100	100	14	0
	16	98	4	0	94	34	0	92	0	0

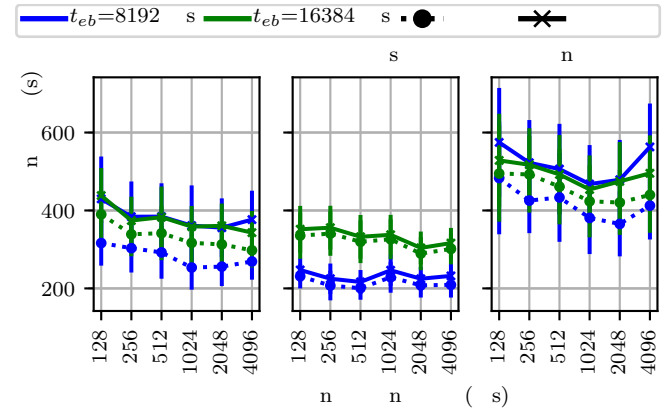


Fig. 3. TSCH synchronisation time and DODAG formation time.

occur than in the ellipse network, primarily when the number of joined nodes increase. Consequently, we notice several desynchronisation events in such topologies, which cause an additional delay after the network-wide TSCH synchronisation. The dynamic of the DODAG formation process in the evaluated scenarios confirms this behaviour. The time, until half of all nodes have joined the DODAG, is approximately the same in all scenarios. In the second half of the network formation, we recognise how the minimal schedule slows down the completion of the process in dense networks.

C. Adding additional links to the 6TiSCH-MC

In this experiment, we investigate the influence of an additional slot in the *minimal schedule* on the time and energy consumed for the network formation. With $N_b = 2$ shared slots in the basic schedule, the 2nd slot will be used, if any frame in the transmission queue is present after the 1st timeslot, but none scheduling algorithm applies. The first remark is that the number of configurations afflicted by unsuccessful network formation decreases. This is an expected behaviour since with a higher value of N_b there is a reduction of local contentions between control packets. Fig. 4 shows a valuable reduction in the average time spent for the network formation. It is also interesting to observe, how the time gap between TSCH synchronisation and DODAG completion shrinks. With $N_b = 2$, there is a more efficient messaging and, therefore, a quicker transition from the state "synchronized" to the state "advertiser" in every joined node.

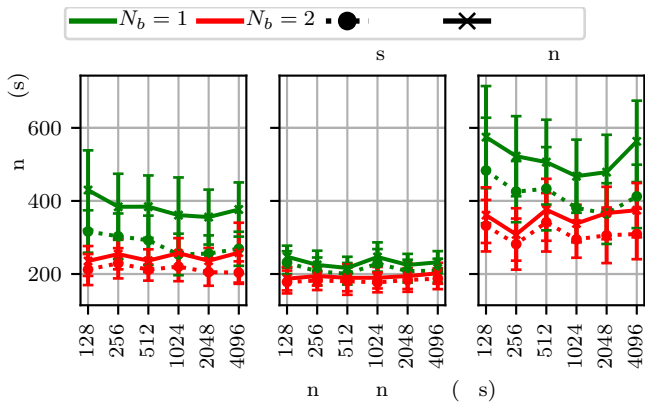


Fig. 4. Time saving in the network formation process by allocating $N_b = 2$ shared slots for the transmission of EBs and RPL messages with $N_c = 4$, $N_{size} = 9$ and $t_{eb} = 8192ms$.

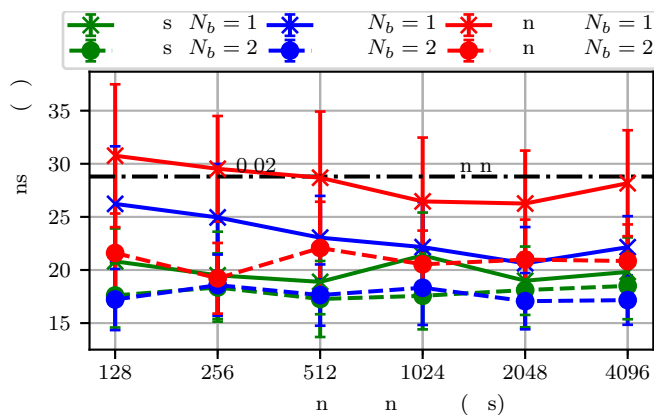


Fig. 5. We assume powering every node (except the sink) with a 2xAA battery pack, which carries 18000C (2x2500 mAh) and we select as parameter set $N_{size} = 9$, $t_{eb} = 8192ms$, $N_c = 4$. We use as current consumption the values inferred from the datasheet of the TI's CC2538 [12], a wireless micro-controller System-on-Chip commonly used for IEEE 802.15.4 applications.

The drawback of adding links to the *minimal schedule* is a higher "basic" duty-cycle since every node is active at least in these broadcast slots. However, the significant reduction of the time spent for network formation causes valuable energy savings, especially in grid and random topologies, as we can see in Fig. 5. This is a consequence of the less time spent in waiting for EB by an activated node and of the reduced number of collisions.

D. Recommendations

This simulation study allows us to derive a set of guidelines on how to choose the TSCH and RPL parameters for a given topology. We recommend implementers of 6TiSCH-network to set the I_{min} parameter slightly over the minimal slotframe duration, while the t_{eb} should be set to a value $t_{eb} \geq 4 \cdot N_s \cdot t_s$, so that the effect of applying randomness to the EB periods can take place. In dense topologies, the number of shared slots used for broadcasting is a crucial factor, and the use of at least $N_b = 2$ is recommended. The duplicated duty-cycle is

compensated by a reduced average joining time of every single node, and by a more likely successful delivery of frame in the whole network.

IV. CONCLUSIONS

We outlined how a low power wireless multi-hop network is formed, i.e., how its presence is announced and how each mote joins to it, as specified by the IETF 6TiSCH WG.

Through simulation, we reported on the limits of the proposed 6TiSCH-MC and on the risk of its blind adoption. Even with small network size, a wrong parameter configuration would cause an unsuccessful synchronisation of nodes within 30 minutes, that means lost operational time and discharged nodes. Among other offered guidelines, we recommend implementers to allocate additional shared slots in the TSCH-schedule responsible for network formation. This approach shows significant time and energy savings during the network formation phase, especially in grid and random topologies. That is a consequence of (1) a reduced number of collisions, (2) a less local contention and (3) a better interplay between TSCH and RPL.

In future works, we plan to simulate with a channel model that implements ray-tracing methods and to experiment with testbeds. These studies need to be carried out in order to validate or reconsider our simulation-based recommendations for more realistic environments. More broadly, we aim at investigating the tradeoff between the network formation time and the duty-cycle of nodes, and at defining a solution for a dynamic, decentralised and collision-avoiding allocation of the $N_b \geq 2$ TSCH-links used for sending broadcasting frames.

REFERENCES

- [1] M. R. Palattella, N. Accettura, X. Vilajosana *et al.*, "Standardized protocol stack for the internet of (important) things," *IEEE communications surveys & tutorials*, vol. 15, no. 3, pp. 1389–1406, 2013.
- [2] *IEEE Standard for Low-Rate Wireless Networks (WPANs)*, IEEE Std. 802.15.4-2015, 2016.
- [3] T. Winter, "Rpl: Ipv6 routing protocol for low-power and lossy networks," RFC 6550, 2012.
- [4] P. Thubert and T. Watteyne. (2013) Ipv6 over the tsch mode of ieee 802.15.4e. [Online]. Available: <https://datatracker.ietf.org/wg/6tisch/about/>
- [5] X. Vilajosana, K. Pister, and T. Watteyne, "Minimal ipv6 over the tsch mode of ieee 802.15. 4e (6tisch) configuration," RFC 8180, 2017.
- [6] D. Fanucchi, B. Staehle, and R. Knorr, "Network formation for industrial iot: Evaluation, limits and recommendations," in *Emerging Technologies and Factory Automation (ETFA), 2018 23rd IEEE International Conference on*. IEEE, 2018, pp. 1–8, in press.
- [7] P. Levis and T. H. Clausen, "The trickle algorithm," RFC 6206, 2011.
- [8] T. Chang, T. Watteyne, K. Pister *et al.*, "Adaptive synchronization in multi-hop tsch networks," *Computer Networks*, vol. 76, pp. 165–176, 2015.
- [9] A. Dunkels. (2002) Contiki: The open source os for the internet of things. [Online]. Available: <http://www.contiki-os.org/>
- [10] L. Wang and A. Reinhardt, "A simulative study of network association delays in ieee 802.15. 4e tsch networks," in *A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2017 IEEE 18th International Symposium on*. IEEE, 2017, pp. 1–3.
- [11] H. Kermajani and C. Gomez, "On the network convergence process in rpl over ieee 802.15. 4 multihop networks: Improvement and trade-offs," *Sensors*, vol. 14, no. 7, pp. 11 993–12 022, 2014.
- [12] TexasInstruments. (2015) Cc2538: A powerful system-on-chip for ieee 802.15.4 applications. [Online]. Available: <http://www.ti.com/product/CC2538/>

Binary Representation of Device Descriptions: CBOR versus RDF HDT

Kristina Sahlmann
University of Potsdam, HTW Berlin
Potsdam, Germany
Email: sahlmann@uni-potsdam.de

Alexander Lindemann
University of Potsdam
Potsdam, Germany
Email: allindem@cs.uni-potsdam.de

Bettina Schnor
University of Potsdam
Potsdam, Germany
Email: schnor@cs.uni-potsdam.de

Abstract—For interoperability, the Internet of Things (IoT) needs self-descriptive devices. We recommend ontology-based device descriptions since the meaning of the information can be extracted. Since the size of these files can be challenging for a constrained device, we have to adapt this approach for the IoT using a suited data representation. Therefore, we evaluate two binary representations: The Concise Binary Object Representation (CBOR) and RDF HDT (Header, Dictionary, Triples).

I. INTRODUCTION

In our previous work [1], [2] we proposed an architecture for a NETCONF-MQTT bridge and self-descriptive IoT devices based on the oneM2M Base ontology [3]. An ontology is a structured vocabulary which describes a certain domain of interest (e.g. metering, sensors, actuators). The ontology defines so-called *triples*:

Subject → **Predicate** → **Object** (*SPO*)

An example for an *SPO* triple within IoT is:

myDevice → **hasFunctionality** → **funcSwitchOff**

Such ontologies, built on triples, represent the semantic meaning of information.

The W3C developed a range of Semantic Web standards. Among them, OWL [4] is defined as a standard for the description of ontologies. OWL is built on the Resource Description Framework (RDF) [5] and has different syntax representations: RDF/XML [6], Turtle [7], N-Triples [8], and JSON-LD [9].

The oneM2M ontology is an OWL ontology and uses RDF/XML syntax by default. Every ontology has its own namespace. Every entity in SPO is represented by an IRI¹ as shown in Listing 1 in N-Triple syntax. However, different syntaxes make the description longer or shorter. An example is given in Listings 2 and 3 which show non-optimized Turtle and optimized JSON-LD.

```
1 <https://www.cs.uni-potsdam.de/bs/research/myno#myDevice>
2 <http://www.onem2m.org/ontology/Base_Ontology/
  base_ontology#hasFunctionality>
3 <https://www.cs.uni-potsdam.de/bs/research/myno#
  funcSwitchOff> .
```

Listing 1. RDF Example in N-Triple syntax

¹<https://www.w3.org/TR/rdf11-concepts/#dfn-iri>

```
1 @prefix owl: <http://www.w3.org/2002/07/owl#> .
2 @prefix oneM2M: <http://www.onem2m.org/ontology/
  Base_Ontology/base_ontology#> .
3 <https://www.cs.uni-potsdam.de/bs/research/myno#myDevice>
4   a owl:NamedIndividual, oneM2M:Device ;
5   oneM2M:hasFunctionality <https://www.cs.uni-potsdam.de/bs/
  /research/myno#funcSwitchOff> .
```

Listing 2. RDF Example in Turtle, non-optimized

```
1 { "@context": {
2   "oneM2M": "http://www.onem2m.org/ontology/Base_Ontology/
  base_ontology#",
3   "myno": "https://www.cs.uni-potsdam.de/bs/research/myno#"
4   },
5   "@graph": [
6     { "@id": "myno:myDevice", ... },
7     "oneM2M:hasFunctionality": [
        { "@id": "myno:funcSwitchOff" }, ...
      ]
    ]
  }
```

Listing 3. RDF Example in optimized JSON-LD syntax

The device description in our use-case scenario comprises descriptions for three functionalities: (i) switch on the LED with a pre-defined color; (ii) switch off the LED; (iii) request the value of the brightness sensor. These capabilities described in the oneM2M ontology result in a file size of 23,966 Bytes using the RDF-XML syntax.

The implementation was done on a CC2538EM microcontroller board of the Development Kit² from Texas Instruments (TI). The TI-Board is a very constrained device with 32 kiB RAM and 512 kiB Flash memory. The TI-Board is also connected by constrained network namely 6LoWPAN [10]. The file size has to be manageable by these resources.

The file size is reduced from 23,966 Bytes (RDF-XML) to 19,771 Bytes by converting it to the JSON-LD syntax. However, the device description will grow again when further services are added. Therefore, we investigate compression possibilities to minimize the file size. Since the compression is performed on a capable compute device before the deployment on the microcontroller board, the compute complexity is not an important criteria and we concentrate only on the file size reduction in our evaluation. We use the same input file in all our experiments.

We can summarize our requirements as follows:

- 1) small file size regarding 32 kiB RAM;

²<http://www.ti.com/tool/cc2538dk>

- 2) ontology data structures particularly strings and structural elements like curly and square braces should be compressed efficiently;
- 3) small count of fragments concerning small MTUs in 6LoWPAN;
- 4) enabling to edit the file on a constrained device e.g. to change the UUID.

The last requirement makes it possible to paste the individual data of a board like the UUID into a generic device description which may be written once for this kind of devices.

There is a variety of compression algorithms. Nevertheless, most of them are not designed for resource-constrained devices. There are few efforts applied to sensor networks e.g. S-LZW [11], SBZIP [12]. We have evaluated CBOR [13], a relatively new standard for the representation of data on constrained devices, and RDF HDT [14] which supports the compression of RDF data and therefore may be well-suited for the binary representation of device descriptions.

This paper is organized as follows: First, related work on CBOR and RDF HDT is presented. Then we introduce both representations briefly and outline the evaluation results. Finally, we discuss the results and give a conclusion.

II. RELATED WORK

CBOR and RDF HDT are relatively new representations, but already investigated by some research groups ([15], [16], [17], [18], [19], [20]).

For the transmission of data within a SIEM system, Rix et al. [15] developed an approach of mapping XML to CBOR using JSON as an intermediate step. Additionally, they applied GZIP compression. By this combination, the example files were compressed from 358 Bytes (XML) to 251 Bytes (XML/GZIP) and to 63 Bytes using CBOR and GZIP resp. 506 (XML), 331 (XML/GZIP) and 141 Bytes (CBOR/GZIP). This shows the benefit of CBOR/GZIP in their use-case scenario.

Pöhls et al. [16] considered CBOR and COSE [21] for the signature of sensor messages in the IoT. Ilgner et al. [17] evaluated CBOR for Bluetooth and 3G Communication to monitor remote pipelines. They used a 8-bit microcontroller (AT Mega 128) with 4 Kbytes of internal static RAM. He experienced some challenges because CBOR consumes a lot of static memory (2.5 Kbytes) for CBOR object variables. CBOR libraries do not support 64-bit integers and longer objects needed to be broken.

RDF HDT was originally developed for the exchange of RDF Datasets in the Web of Data. A comprehensive evaluation on HDT was done by Fernandez et al. [18], [19] and Martinez et al. [20]. They show examples where the HDT files took only between 6% and 11% space compared to the original N-Triples³. This compression results are impressive and make HDT an interesting candidate.

III. CBOR AND JSON

The Concise Binary Object Representation (CBOR) is defined in RFC 7049 [13] in 2013. The next version is still work-

³<http://www.rdfhdt.org/hdt-internals/>

in-progress [22]. The goal of CBOR is a standardized format for binary representation of structured data. The conversion from JSON to CBOR and vice versa was defined as part of RFC 7049. Objectives of CBOR are the representation of basic data types and structures of JSON using binary encoding, a compact encoder and decoder code supporting constrained devices, and self-describing data so that a generic decoder can be written.

A. CBOR Encoding

With CBOR encoded data are self-describing and can be decoded without a schema description. CBOR defines 8 major types (see Table I).

TABLE I
CBOR MAJOR TYPES

	Major Type
0	unsigned integer
1	negative integer
2	byte string
3	text string
4	array of data items
5	map of pairs of data items
6	optional semantic tagging of other major types
7	floating-point numbers and simple data types

Each byte is encoded as a major type (the high-order 3 bits) and additional information (the low-order 5 bits) (see Table II).

TABLE II
INITIAL BYTE OF EACH DATA ITEM IN CBOR

X X X	X X X X X
major type	additional information

The resulting integer in all additional information values is interpreted depending on the major type. For example, it represents the integer value itself for integer type if the value of additional information is less than 24. The integer value 20 is encoded as *0x14* or binary *000 10100* with major type 0 and additional information 20.

The byte string "CBOR" is encoded as *0x6443424F52* or binary *011 00100 01000011 01000010 01001111 01010010* with major type 3, additional information 4 (string length) and the string value in bytes. An implementation of a CBOR decoder can use a jump table with all 256 defined values for the initial byte.

B. Evaluation

For the evaluation of CBOR, we implemented a CBOR2JSON decoder and JSON2CBOR encoder in Python which is based on the CBOR library⁴. The conversion of the ontology file from RDF/XML format to JSON-LD is done by Protégé⁵. This file is still well-formed for human-reading i.e. includes white spaces and tabs.

⁴<https://github.com/agronholm/cbor2>

⁵<http://protege.stanford.edu/>

TABLE III
ONTOLOGY FILE SIZE IN BYTES FOR JSON-LD AND CBOR

JSON-LD	19,771
CBOR	16,126
CBOR to JSON	17,142

TABLE IV
ONTOLOGY FILE SIZE IN BYTES FOR JSON-LD (OPT.) AND CBOR

JSON-LD (opt.)	7,640
CBOR	6,520
CBOR to JSON-LD	7,640

The results of the conversion are shown in Table III. The third row shows the file size after the reverse conversion. CBOR achieved 18.43% space savings applied to non-optimized JSON-LD. The major drawback of CBOR is the poor compression of strings. Strings up to 23 Bytes length can save one Byte per string. Strings up to 256 Bytes length have no savings, and longer strings require one Byte more per string [23].

Next, we optimized the JSON-LD file by two steps: (1) We introduced context (with namespaces) for compact IRIs using JSON-LD Playground⁶; (2) we removed white spaces and tabs which are unnecessary for machine-reading. The conversion results of the optimized JSON-LD to CBOR representation are shown in Table IV. Obviously, optimization of JSON-LD file pays off because the strings are shortened. CBOR achieves still 14.65% space savings compared to the optimized JSON-LD file.

Thus, optimized JSON-LD and CBOR reduce the file size according to the requirements 1, 2 and 3. The editing of the CBOR file requires more effort to fulfill the requirement 4 because we have to replace a string in a binary encoded file. Therefore, we implemented a function `CBORStreamInject` in C using key-value pairs in an additional meta file. Based on a key, the position will be found in the file and the value inserted.

IV. RDF HDT

RDF HDT is a W3C Member Submission [14] from 2011. HDT was developed to share big semantic datasets in RDF format on the web. The data are compressed to save space but enable search and browse operations without decompression. HDT addresses high levels of verbosity and redundancy in RDF files and machine-understandability.

A. HDT Encoding

Technically, as its name implies, HDT splits RDF data in three logical components: Header, Dictionary and Triples. The Header holds metadata about the file using a plain RDF structure. The Dictionary is a catalog of triples for the used terms such as IRIs in an ontology. The dictionary encoding is based on the Front-Coding [24] which is again based on words with similar prefix. Subjects, Predicates and Objects

⁶<https://json-ld.org/playground/>

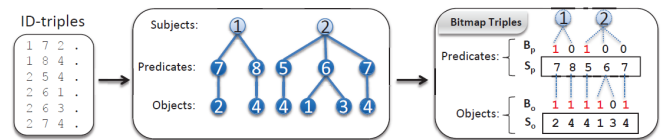


Fig. 1. Description of Bitmap Triples [20]

TABLE V
ONTOLOGY FILE SIZE IN BYTES FOR TURTLE AND RDF HDT

Turtle	20,385
HDT	5,568
HDT to Turtle	17,244

are subsumed in sections to avoid repeating. Each section is sorted lexicographically and then IDs are assigned to each term. RDF Triples are now tuples of three IDs. HDT proposes an encoding of Triples called Bitmap Triples (BT). SPO trees are sorted in a specific order like many trees in a forest and a binary encoding is applied layer by layer, as shown in Figure 1.

B. Evaluation

For the conversion, we use the HDT C++ Library⁷. This library accepts Turtle or N-Triples syntax of ontology files. Beforehand, we converted our device description from RDF/XML to Turtle using Protégé. Notice, there was no optimization e.g. removing unnecessary white spaces and tabs. The conversion results are shown in Table V. Then we converted the optimized JSON-LD file to Turtle using the Easy RDF converter⁸ (see Table VI). Finally, we converted the ontology file to N-Triples using Protégé. The corresponding results are given in Table VII. The conversion from N-Triples results in the smallest file because of the string syntax. The N-Triples representation uses only quotation marks for strings, and Turtle adds the data type `http://www.w3.org/2001/XMLSchema#` string to every string.

We observe space savings of 72.68% in Table V and 84.06% in Table VII because the input files have a verbose syntax. In both cases of the Turtle compression, the resulting HDT file is nearly identical (5,568 versus 5,513 Bytes). Only the Dictionary part is shorter when using optimized Turtle as input. The reason of only 44.82% space savings in Table VI is the optimized Turtle where the IRIs are shorten by prefix. The file grows by reverse converting from HDT to Turtle because the software uses the same algorithm and produces a verbose file with full IRIs.

HDT addresses our requirements 1, 2 and 4 regarding small file size and RDF data structure. The requirement 3 about editing the file without prior decompression is only partly fulfilled: it is possible to exchange the string values in the Dictionary part (e.g. change a UUID) but side effects may occur because the meta data in the Header section doesn't match anymore (i.e. `dictionarysizeStrings`).

⁷<https://github.com/rdfhdt/hdt-cpp>

⁸<http://www.easyrdf.org/converter>

TABLE VI
ONTOLOGY FILE SIZE IN BYTES FOR TURTLE (OPT.) AND RDF HDT

Turtle (opt.)	9,992
HDT	5,513
HDT to Turtle	17,244

TABLE VII
ONTOLOGY FILE SIZE IN BYTES FOR N-TRIPLES

N-Triples	27,025
HDT	4,307
HDT to N-Triples	26,848

V. DISCUSSION

In our tests, CBOR showed a poor compression rate compared to JSON-LD files. Additionally, the requirements 2 and 4 are only rudimentary supported by CBOR. A more detailed evaluation of CBOR is given in [23]. Some tests in [23] result only by 8.9% space savings. The recommendation of this evaluation is: do not exceed the length of 23 bytes for strings; replace strings by integer and literals when possible.

RDF HDT achieved better compression results compared to RDF Turtle and N-Triples files. Moreover, HDT files are smaller than CBOR files. Contrary to CBOR, HDT applies native RDF algorithms considering data structure and long strings. However, HDT has still some restrictions to edit the file due to the encoding algorithm: meta data should not be modified in the Header component; string values can be modified in the Dictionary; but it is not possible to edit the Triples component.

VI. CONCLUSION

We evaluated two binary representations, CBOR and RDF HDT, for ontology-based device-descriptions. HDT has shown better space savings than CBOR in our use-case. HDT is constructed for ontology files in RDF format and therefore optimized on this structure. CBOR has a straightforward approach applied to data types without considering the data structure. CBOR is not well suited for the compression of ontologies, since long strings, which are the main part in device descriptions, are not efficiently compressed (less than 15 % savings in our example). But CBOR may still have its strength when sensor data have to be encoded for transmission.

Overall, RDF HDT is a promising candidate for further evaluation on constrained devices.

ACKNOWLEDGMENT

This research is funded by the Federal Ministry of Education and Research under grant number Professors Program II. The responsibility for the content of this publication lies with the authors.

REFERENCES

[1] K. Sahlmann, T. Scheffler, and B. Schnor, "Managing IoT device capabilities based on oneM2M ontology descriptions," in *Proceedings of the 16. GI/ITG KuVS Fachgespräch Sensornetze*, ser. Technical Reports. Hamburg, Germany: HAW Hamburg, 2017, pp. 23–26.

[2] —, "Ontology-driven Device Descriptions for IoT Network Management," in *IEEE Global Internet of Things Summit (GloTS) Proceedings, InterOSS-IoT 2018*. Piscataway, NJ, USA: IEEE, 2018, pp. 353–358, accepted.

[3] "oneM2M Technical Specification: TS-0012-V2.0.0," 2016. [Online]. Available: www.onem2m.org/images/files/deliverables/Release2/TS-0012-oneM2M-Base-Ontology-V2_0_0.zip

[4] "OWL 2 Web Ontology Language Document Overview (Second Edition)," W3C, Recommendation, Dec. 2012.

[5] R. Cyganiak, M. Lanthaler, and D. Wood, "RDF 1.1 Concepts and Abstract Syntax," W3C, Recommendation, Feb. 2014.

[6] F. Gandon and G. Schreiber, "RDF 1.1 XML Syntax," W3C, Recommendation, Feb. 2014.

[7] G. Carothers and E. Prud'hommeaux, "RDF 1.1 Turtle," W3C, Recommendation, Feb. 2014.

[8] A. Seaborne and G. Carothers, "RDF 1.1 N-Triples," W3C, Recommendation, Feb. 2014.

[9] M. Lanthaler, G. Kellogg, and M. Sporny, "JSON-LD 1.0," W3C, Recommendation, Jan. 2014.

[10] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks," Internet Requests for Comments, IETF, RFC 4944, September 2007.

[11] C. M. Sadler and M. Martonosi, "Data Compression Algorithms for Energy-Constrained Devices in Delay Tolerant Networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*, ser. ACM Digital Library. New York, NY: ACM, 2006.

[12] N. Tsiftes, A. Dunkels, and T. Voigt, "Efficient Sensor Network Reprogramming through Compression of Executable Modules," in *5th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, 2008, pp. 359–367.

[13] C. Bormann and P. Hoffman, "Concise Binary Object Representation (CBOR)," Internet Requests for Comments, IETF, RFC 7049, October 2013.

[14] J. D. Fernández, M. A. Martínez-Prieto, C. Gutierrez, and A. Polleres, "Binary RDF Representation for Publication and Exchange (HDT)," 2011. [Online]. Available: <https://www.w3.org/Submission/HDT/>

[15] T. Rix, K.-O. Detken, and M. Jahnke, "Transformation between XML and CBOR for network load reduction," in *3rd International Symposium on Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS)*, 2016, pp. 106–111.

[16] H. C. Pöhls and B. Petschkuhn, "Towards compactly encoded signed IoT messages," in *IEEE 22nd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2017, pp. 1–6.

[17] H. Ilgner and S. Pienaar, "Implementing a Compact Data Format for Bluetooth and 3G Communication to Monitor Remote Pipelines," in *International Conference on Advances in Computing and Communication Engineering (ICACCE)*, 2016, pp. 45–50.

[18] J. D. Fernández, M. A. Martínez-Prieto, C. Gutiérrez, A. Polleres, and M. Arias, "Binary RDF Representation for Publication and Exchange (HDT)," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 19, p. 2241, 2013. [Online]. Available: <http://www.websemanticsjournal.org/index.php/ps/article/view/328>

[19] J. D. Fernández, M. A. Martínez-Prieto, and Polleres, Axel and Reindorf, Julian, "HDTQ: Managing RDF Datasets in Compressed Space," in *15th Extended Semantic Web Conference (ESWC) 2018*, 2018.

[20] M. A. Martínez-Prieto, M. Arias, and J. D. Fernández, "Exchange and Consumption of Huge RDF Data," in *The Semantic Web: Research and Applications*. Springer, 2012, pp. 437–452.

[21] J. Schaad, "CBOR Object Signing and Encryption (COSE)," Internet Requests for Comments, IETF, RFC 8152, July 2017.

[22] C. Bormann and P. E. Hoffman, "Concise Binary Object Representation (CBOR): Internet-Draft." [Online]. Available: <https://datatracker.ietf.org/doc/html/draft-ietf-cbor-7049bis-02>

[23] C. Hoffmann, "Effiziente Datendarstellung mittels CBOR für das Internet of Things," Bachelor's Thesis, University of Potsdam, Germany, May 2018.

[24] I. H. Witten, A. Moffat, and T. C. Bell, *Managing gigabytes: Compressing and indexing documents and images*, 2nd ed. San Francisco, Calif.: Kaufmann, 2007.

A Case for Chirp Modulation for Low-Power Acoustic Communication in Shallow Waters

Jan Heitmann, Fabian Steinmetz, Christian Renner

Research Group smartPORT

Hamburg University of Technology

Email: {jheitmann, fabian.steinmetz, christian.renner}@tuhh.de

Abstract—Small and cheap micro AUVs enable diverse underwater monitoring applications in shallow inshore waters; e.g., inspection of underwater assets, observation of water quality, and identification of pollution sources. The formation and collaboration of swarm members yet requires communication and self-localization based on cheap, miniature acoustic devices. However, this is severely hampered by the effects of multi-path propagation in shallow waters. We study the benefits and applicability of narrow-band chirp-based modulation vs. frequency-shift keying (FSK). First, we discuss the influence of multi-path propagation on FSK through real-world experiments. Second, we show that narrow-band chirps can be used as a direct replacement of FSK symbols, increasing the detection probability by up to 23% and accuracy and precision by up to 39% and 53%, respectively, at virtually no bandwidth penalty.

I. INTRODUCTION

Exploration and monitoring of underwater sceneries is drawing considerable attention. Recent examples are the investigation of sub-mesoscale eddies [1] or ship tracking in harbors [2]. Timely acquisition of data mandates communication—typically wireless to keep installation and maintenance cost low. Acoustic underwater modems have been proposed by academia [3], [4] and launched by industry [5]–[7] for this purpose.

Shallow and relatively small water bodies—such as port basins, lakes, or canals—are a particularly challenging scenery. Reflections at the water surface cause massive inter- and intra-symbol interference, because the non-line-of-sight (NLOS) signals have a low delay and attenuation only [8]. In previous research [3] and recent measurement campaigns, we experienced significant attenuation and amplification of frequency shares depending on position and time. Figure 1 indicates the frequency selectivity of the acoustic channel. In addition, we noted influence of environmental conditions and their change over time. Proposed countermeasures are, e.g., frequency hopping, PSK, and OFDM [9], [10]. Unfortunately, inshore applications and the implied use of μ AUVs mandates miniature, low-power modems such as [3] with low computing power and resources that are insufficient to run such algorithms.

We embark on a different strategy, namely the use of narrow-band chirp signals for preamble-based synchronization, motivated by the following observations. We noted that communication with our acoustic modem was—despite using FSK modulation—reliable, if the preamble was successfully detected by the receiver. Failure to do so arises mainly from

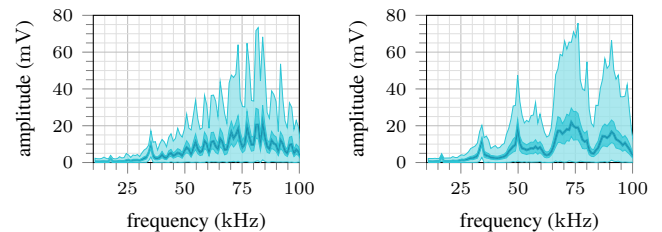


Fig. 1: Quartiles of received frequency shares of a 1s band-limited (10 kHz to 100 kHz) pseudo-random acoustic signal at 2.1 m (left) and 5.1 m (right). Received signals were sampled at 1 MHz and broken into 976 segments of 1024 samples each. Sender and receiver were submerged ~ 0.5 m in LOS conditions in a marina in Hamburg.

cancellation and amplification caused by reflections and scattering. The main reason for this is intra-symbol interference leading to unreliable symbol detection and hence synchronization. Once the synchronization has succeeded, however, the symbol windows are known and reliable communication can be achieved through relatively simple methods such as frequency hopping and redundancy coding.

We make the following contributions: We motivate the benefit of narrow-band chirp modulation for preamble-based synchronization. We sketch a low-complexity implementation for low-power acoustic modems. We evaluate our approach against two non-coherent detection and cross-correlation for FSK modulation. Our results are based on real-world experiments in conditions similar to typical inshore, shallow-water application scenarios.

II. FUNDAMENTALS

A. Frequency-Shift Keying

Frequency-Shift Keying (FSK) is commonly used in acoustic underwater communication. In binary frequency-shift keying (BFSK), a bit b is transmitted as a sinusoidal symbol with frequency f_b and duration T . There are more complex forms of FSK, which we do not address in detail due to space constraints.

Receivers often employ non-coherent detection due to its efficiency, but cross-correlation is also possible. Orthogonal frequencies f_b may improve detection. The (envelope) shape of detector output is trapezoidal for undistorted symbols with constant amplitude, with its peak marking the symbol's end. If reflections overlap with the line-of-sight (LOS) signal, detector

output contains overlapping triangles as in Figs. 2a and 2b. Depending on the number and phase of reflections, a clear peak is no longer present, impacting detection accuracy and likelihood.

B. Chirp Keying

A chirp is a signal with steadily changing frequency over time. We consider linear chirps, where the frequency sweeps linearly over time from frequency f_s to f_e . The chirp has duration T and bandwidth $B = |f_e - f_s|$. A bandwidth-efficient way to employ binary modulation is to use down-chirps ($f_s > f_e$) and up-chirps ($f_s < f_e$) to represent bit b .

Detection of a chirp can be achieved through cross-correlation. The detector output is a narrow and steep peak, which allows a clear distinction between the LOS signal and reflected signals as displayed in Fig. 2c. Because of this property, chirps are commonly employed in radar applications. For a given T , B can be chosen such that up- and down-chirp are orthogonal, or vice versa.

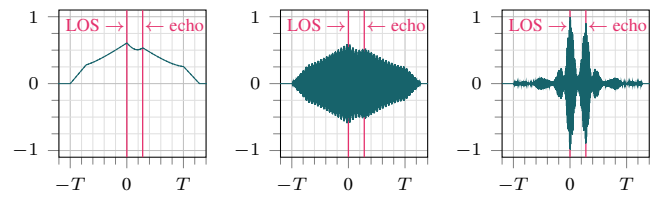
C. Comparison and Discussion

In various real-world experiments (cf. Fig. 1 and [3]), we confirmed heavy, time-varying, and location-dependent frequency selectivity of the acoustic channel. This is of paramount relevance for preamble-based synchronization. Here, symbol (window) positions are derived by detecting the preamble symbols or their peaks, respectively. For successful packet reception and precise time-of-flight ranging [11], accurate synchronization is mandatory. In case of FSK, signal cancellation and reverberation due to reflections cause poor synchronization. As shown in Fig. 2, the peak of the originally trapezoidal detection result is blurred and accurate detection hampered. Noise and multiple echos aggravate this situation. In case of destructive interference, a peak may not be found at all.

Chirp modulation brings two advantages: (i) its frequency spreading increases detection probability, because attenuation of frequency shares due to reflection is drastically reduced, and (ii) the peaks of the LOS signal and its echos are separable in the time domain, so that a smaller synchronization error is expected. Using chirps yet comes at a cost. The benefit of non-coherent FSK detection is its low computation complexity. For each new sample, two multiplications are required, giving a constant computation complexity per new sample. Without optimization, cross-correlation has to be performed for a full symbol for every sample, yielding at least logarithmic computation complexity per new sample. For low-power acoustic modems, this complexity may already be infeasible. Hence, we study the performance of chirp vs. FSK in general but also discuss the feasibility of chirp detection on low-power acoustic modems.

III. PREAMBLE-BASED SYNCHRONIZATION

Preamble-based synchronization is achieved through a preamble of N_{pre} symbols of alternating up- and down-chirps followed by a start frame delimiter (SFD) of N_{sfd} symbols.



(a) FSK non-coherent (b) FSK cross-corr. (c) chirp cross-corr.

Fig. 2: Detection of chirp ($f_s = 50$ kHz, $f_e = 54$ kHz) and FSK ($f = 50$ kHz) signal superimposed by an echo with 0.7 ms delay and 90% amplitude (ca. 10 m distance at 2 m depth). Symbol duration is $T = 2.5$ ms. Time on the x-axis is relative to perfect LOS detection.

To detect the presence of a synchronization sequence, the receiver employs a signal level threshold τ . The reason for this approach is that continuous cross-correlation for each new sample is typically infeasible on low-power modems.

To detect the first preamble symbol after threshold exceedance, a cross-correlation over a window of length $2T$ is calculated. We chose twice the symbol duration to obtain an extended window to search for correlation peaks. The highest peak is accepted as valid preamble symbol, if its value exceeds the mean of the correlation values by a factor of ρ . After detection of the first peak, windows of length $(1 + \alpha)T$ ($0 < \alpha < 2$) in steps of T relative to the first symbol (peak) are used to detect the remaining preamble symbols. Note that factor α sets the size and overlap of the search windows of the individual symbols and marks a trade-off between calculation speed and detection tolerance. When the SFD sequence is identified, we take the median of all stored detection times, delete all synchronization symbols with a deviation of more than βT ($0 < \beta < 1$). The mean value of the remaining peaks' times is finally accepted as synchronization point—i.e., the symbols of the following data packet are expected in intervals of T relative to this time.

Based on previous finding and dedicated experiments, we chose $N_{\text{pre}} = 16$ and $N_{\text{sfd}} = 4$, $\alpha = 0.4$, $\tau = 20\%$, $\rho = 3$, and $\beta = 0.1$.

IV. EXPERIMENTATION SETUP

For all experiments, we used the smartPORT acoustic modem [3], a low-power, low-cost device for use in μ AUVs such as MONSUN [12] and Hippocampus [13]. The required amount of signal processing limits the sampling frequency to 200 kHz. The receive circuitry embraces an analog 16th-order bandpass filter with passband from 50 kHz to 75 kHz and a digitally-adjustable gain. The communication range is above 100 m with an Aquarian Audio AS-1 hydrophone as transducer.

For evaluation of our chirp-based synchronization, we conducted real-world tests in the marina of TuS Finkenwerder, which offers ideal and realistic testing conditions: Several jetties of up to 150 m length allow for testing at versatile distances and in LOS conditions. This plot resembles the envisioned scenario in rivers and port basins. The water is at least 2 m deep.

Due to the fundamental nature of our investigation, signal preparation and processing was done with MATLAB. Signal generation and recording was performed with a TiePie HS5 USB oscilloscope and waveform generator connected to the receiver and the transmitter circuitry of our acoustic modem. Sender and receiver were connected to the same oscilloscope to enable calculation of absolute time differences. Cable lengths limited the maximum distance to 5.1 m.

Unless stated otherwise, the sampling rate of the receiver was 200 kHz to match the setting of our modem. Hydrophones were placed in a depth of ca. 0.5 m. To obtain a ground-truth for synchronization accuracy, we sent a long, wide-band chirp (10 kHz to 150 kHz, 100 ms) sampled with 1 MHz. For each distance, this measurement was repeated 10 times, producing a reference value with a standard deviation of at most 2.1 μ s.

V. RESULTS

A. Synchronization Macroscope

First, we evaluated chirp vs. FSK synchronization on a wide frequency band of 40 kHz to 90 kHz with frequency steps of 2 kHz. The distance between sender and receiver was 5.1 m. We performed 100 synchronization attempts per modulation scheme and frequency with $T = 2.5$ ms and $B = 2.342$ kHz, which produces orthogonal up- and down-chirps. We examined synchronization accuracy and precision in terms of the mean and standard deviation of the absolute synchronization errors. Besides the LOS signal, we observed a dominant (surface) reflection arriving with a mean delay of 54 μ s (cf. Sect. IV).

Figure 3 shows heatmaps of the synchronization errors. The figure reveals high variance for non-coherent (Fig. 3a) and cross-correlation (Fig. 3b) FSK; i.e., lower precision compared to chirps (Fig. 3c). Only in rare cases, the benefit of chirp modulation is small—e.g., for $f = 90$ kHz, precision is lower than for FSK; and for $f = 46$ kHz, its benefit is marginal.

An important aspect of synchronization is the percentile of successful detection within a given margin of a symbol duration T , because this will affect the number of bit errors during demodulation of the transmitted data packet; e.g., a 10% margin means that the actual symbol and demodulation window overlap by 90%. According to our measurements, chirping achieves 83.8% vs. 76.8% and 67.9% for cross-correlation and non-coherent FSK, respectively, for a margin of 10% (or 250 μ s). This implies a 23% improvement of chirping over non-coherent FSK. For a 20% margin, these numbers climb up to \sim 90% for all modulation schemes.

Regarding accuracy and precision, chirping outperforms non-coherent FSK with an average synchronization error of 56.5 μ s vs. 93.1 μ s and a standard deviation of 39.9 μ s vs. 84.9 μ s for a 10% margin. Surprisingly, cross-correlation FSK has better (average) accuracy (34.5 μ s) but lower precision (99.0 μ s). Figure 3b shows many early synchronizations, which are caused by received signal cancellation due to reflections. In case of chirping, on the contrary, there is a tendency to late detection caused by the surface reflection. In addition to accuracy, we also calculated absolute synchronization errors, because from a perspective of successful packet reception,

negative and positive errors do not cancel out from packet to packet. The corresponding numbers are 75.2 μ s for chirp, 106.3 μ s for cross-correlation, and 119.9 μ s for non-coherent FSK, exhibiting a 29.3% advantage of chirping. Values for a 20% margin are comparable and omitted.

B. Synchronization Microscope

We further explored the previous findings by examining the distribution of synchronization times for one frequency band in detail. For this experiment, 1000 synchronization sequences have been sent for each modulation scheme and at distances of 2.1 m and 5.1 m. We used $f_0 = 62.44$ kHz, $f_1 = 62.84$ kHz for FSK and $f_s = 62.44$ kHz, $f_c = 64.782$ kHz for chirping.

Figure 4 shows the distribution of synchronization errors. At a distance of 2.1 m, chirping has a 13.7% higher success rate compared to FSK, whereas the improvement of accuracy and precision is small. However, at a distance of 5.1 m (Figs. 4a to 4c), chirp synchronization improves overall success rate of up to 20.7% and also accuracy and precision are improved. The reason for this is that at larger distances, LOS and NLOS (surface reflection) signals exhibit less separation in time-domain. Figure 5 provides a detailed study of our performance metrics, and it shows that chirping improves the synchronization precision up to factor of 3.1. It is also evident that chirping elevates the success rate considerable and provides higher accuracy.

VI. CONCLUSION

Underwater communication and self-localization are required for the deployment of swarms of μ AUVs. Preamble-based synchronization for communication is hampered due to reflections in shallow waters. We showed that chirp modulation for synchronization is generally superior to FSK modulation, as it improves accuracy, precision, and success rate. This enables higher data rates, less packet loss, and better (packet-based) localization accuracy. We experienced that the benefit of using chirp synchronization is larger at a distance of 5.1 m than at 2.1 m. We expect that this also holds for much larger distances, because the relative error between LOS signal and first echo decreases at large distances. However, chirp detection is more complex and needs carefully designed algorithms and implementations for low-power acoustic modems.

ACKNOWLEDGMENT

This work has been partially supported by the German Federal Ministry of Education and Research (BMBF, FKZ 13N14153), the German Federal Ministry for Economic Affairs and Energy (BMW, FKZ 03SX463C), and ERA-NET Cofund MarTERA (contract 728053). The authors would like to thank Lucas Bublitz for building acoustic modems and TuS Finkenwerder for granting access to their jetties and facilities.

REFERENCES

- [1] B. Meyer, C. Isokeit, E. Maehle, and B. Baschek, "Using Small Swarm-Capable AUVs for Submesoscale Eddy Measurements in the Baltic Sea," in *Proc. of the MTS/IEEE Oceans Conf.*, Sep. 2017.

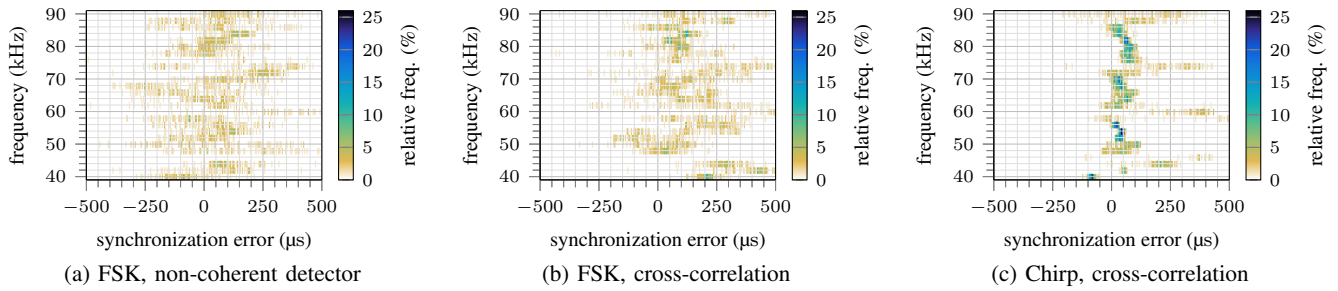


Fig. 3: Heatmaps of synchronization errors for chirp and FSK-based synchronization and 5.1 m LOS distance. Time resolution is 5 μ s.

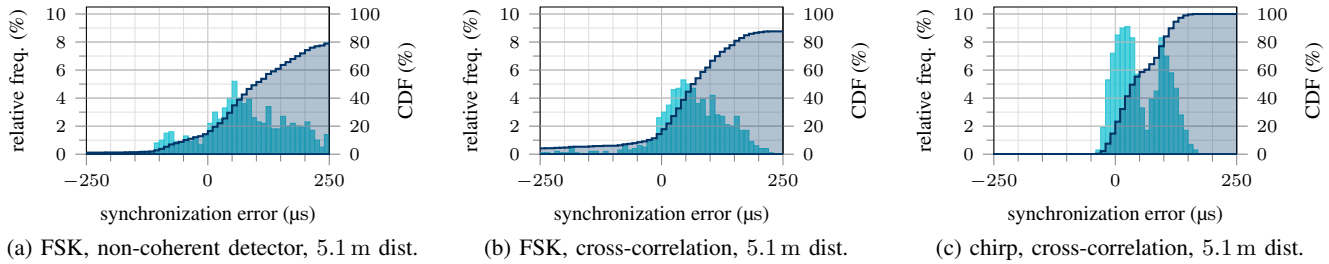


Fig. 4: Distribution and cumulative distribution function (CDF) of synchronization errors for chirp and FSK based synchronization ($f_0 = 62.44$ kHz, $f_1 = 62.84$ kHz, $f_s = 62.44$ kHz, $f_e = 64.782$ kHz). Turquoise graphs show the relative distribution, blue line the cumulative frequencies. Failed synchronizations or this with an error of more than ± 250 μ s are not included in the plot for better readability.

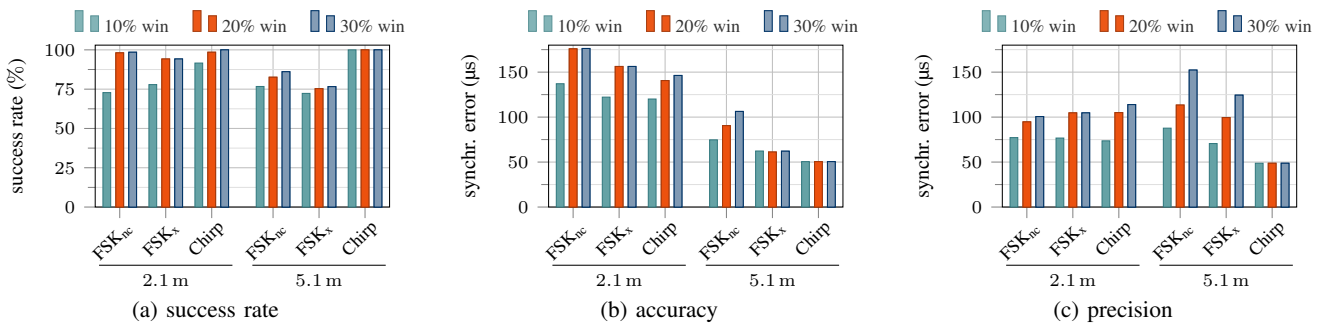


Fig. 5: Comparison of success rate, accuracy, and precision of synchronization with non-coherent (FSK_{nc}) and cross-correlation (FSK_x) vs. chirping. Grouped bars show values for different margins (in percent) w.r.t. symbol length; e.g., a 10% margin is equivalent to 250 μ s.

- [2] M. Rao, N. K. Kamila, and K. V. Kumar, "Underwater Wireless Sensor Network for Tracking Ships Approaching Harbor," in *Proc. of the IEEE International Conf. on Signal Processing, Communication, Power and Embedded System (SCOPE5)*, Oct. 2016.
- [3] C. Renner and A. J. Golkowski, "Acoustic Modem for Micro AUVs: Design and Practical Evaluation," in *Proc. of the 11th ACM International Conf. on Underwater Networks & Systems (WUWNet)*, Shanghai, China, Oct. 2016.
- [4] E. Gallimore, J. Partan, I. Vaughn, S. Singh, J. Shusta, and L. Freitag, "The WHOI Micromodem-2: A Scalable System for Acoustic Communications and Networking," in *Proc. of the MTS/IEEE Oceans Conf.*, Sep. 2010.
- [5] Evologics GmbH, "Underwater Acoustic Modems," <http://www.evologics.de/en/products/acoustics/>.
- [6] Trittech International Limited, "Micron Data Modem – Acoustic Modem," <http://www.tritech.co.uk/product/micron-data-modem>.
- [7] Teledyne Benthos, "ATM-903 series (OEM)," http://teledynebenthos.com/product/acoustic_modems/903-series-atm-903.
- [8] T. Jensenud and S. Ivansson, "Modeling the Power Delay Profile of Underwater Acoustic Channels — The Effects of Out-of-Plane Scattering and Reverberation," in *Proc. of the International Conf. Underwater Communications and Networking (UComms)*, Sestri Levante, Italy, Sep. 2014.
- [9] X. Zhao, D. Pompili, and J. Alves, "Energy-efficient OFDM Bandwidth Selection for Underwater Acoustic Carrier Aggregation Systems," in *Proc. of the International Conf. Underwater Communications and Networking (UComms)*, Lercici, Italy, Aug. 2016.
- [10] K. Pelekanakis, L. Cazzanti, G. Zappa, and J. Alves, "Decision Tree-based Adaptive Modulation for Underwater Acoustic Communications," in *Proc. of the International Conf. Underwater Communications and Networking (UComms)*, Lercici, Italy, Aug. 2016.
- [11] C. Renner, "Packet-Based Ranging with a Low-Power, Low-Cost Acoustic Modem for Micro AUVs," in *Proc. of the 11th International ITG Conf. on Systems, Communications and Coding (SCC)*, Hamburg, Germany, Feb. 2017.
- [12] B. Meyer, C. Renner, and E. Maehle, "Versatile Sensor and Communication Expansion Set for the Autonomous Underwater Vehicle MONSUN," in *Proc. of the 19th International Conf. on Climbing and Walking Robots and Support Technologies for Mobile Machines (CLAWAR)*, London, UK, September 2016.
- [13] A. Hackbarth, E. Kreuzer, and E. Solowjow, "HippoCampus: A Micro Underwater Vehicle for Swarm Applications," in *Proc. of the IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015.

Test-Framework zur softwarebasierten Fehlerinjektion, -stimulation und Protokollierung von WSN-Anwendungen

Max Frohberg, Sebastian Reinhold, Paul Poppe, Mario Schölzel

IHP

Im Technologiepark 25

Frankfurt (Oder), Germany

Email: {frohberg, reinhold, poppe, schoelzel}@ihp-microelectronics.com

Zusammenfassung—Tests und Verifikation sind ein fundamentaler Bestandteil im Softwareentwicklungsprozess. Im Bereich der Drahtlosen Sensornetze ist ein intensives Testen sowohl der Anwendung als auch des Gesamtsystems unabdingbar. Aufgrund ihrer physischen Erreichbarkeit sind diese nur schwer zu debuggen. Insbesondere bei der Entwicklung von Kommunikationsprotokollen kann das Verhalten im Feld unter realen Bedingungen vom zuvor simulierten abweichen. In diesem Paper wird ein Framework zur softwarebasierten Fehlerinjektion im Bereich von WSN-Anwendungen auf Grundlage einer Cross-Plattform (CP) vorgestellt. Die CP unterstützt die Portierung einer Anwendung auf eine andere Plattform durch Nutzung unterschiedlicher Betriebssysteme. Nach der Portierung wird diese mittels Fehlerinjektion und Manipulation der Datenströme getestet. Ein zusätzlicher Mechanismus zum Protokollieren unterstützt zudem die Fehlersuche. In Kombination ist somit die Eingrenzung oder sogar die Rekonstruktion der Fehlerursache möglich. Das hier vorgestellte Framework wurde für Einzel- und Netzwerktests in einem WSN und insbesondere für die Entwicklung von MAC- und Routing-Protokollen konzipiert. Es eröffnet zudem die Möglichkeit, einfache Peripherie wie Sensoren oder Aktuatoren zu emulieren, falls diese auf der Entwicklungsplattform nicht zur Verfügung stehen.

I. EINLEITUNG

Fehler in der Softwareentwicklung haben zum Teil fatale Auswirkungen und verursachen umso mehr Kosten, je später sie gefunden werden [1]. Ein modelgetriebenes Vorgehen kann den Entwicklungsprozess dabei unterstützen, sodass Fehler möglichst vermieden werden können. So wird beispielsweise beim V-Modell jeder Phase im Entwicklungsprozess eine Testphase zur Qualitätssicherung zugeordnet und deren Ergebnis mit den jeweiligen Anforderungen abgeglichen. Je weiter der Entwicklungsprozess voranschreitet, umso konkreter wird die eigentliche Implementierung. Fehler die hier nicht oder zu spät entdeckt werden, haben in der Regel ein schwieriges und zeitaufwändiges Debuggen der Software zur Folge.

Bei der Entwicklung im Bereich von Drahtlosen Sensornetzen (WSN) kommen zudem noch weitere Herausforderungen hinzu. So kommen zum Beispiel Cross-Compiler zum Einsatz, da Software in der Regel auf einer anderen Plattform entwickelt wird als sie später eingesetzt werden soll. Einzelne Sensorknoten können zudem über eine große Fläche verteilt und nur schwer erreichbar sein. Auch hat die energieeffiziente

Entwicklung von Kommunikationsprotokollen maßgeblichen Einfluss auf die Langlebigkeit einer Applikation, da nur begrenzte Energieressourcen zur Verfügung stehen [2]. Des Weiteren muss die Software auch Fehler in der Hardware oder Peripherie erkennen und behandeln und für einen zuverlässigen Betrieb daher hinreichend getestet werden.

Im Bereich der WSN ist es im Sinne der Einsatzdauer sinnvoll, auch den Stromverbrauch zu messen und schon während der Umsetzung konsequent zu beobachten und zu bewerten. Zudem stehen zahlreiche Simulationswerkzeuge wie *OM-Net++*, *Cooja* unter *ContikiOS* oder *TOSSIM* unter *TinyOS* zur Verfügung, die je nach zugrundeliegendem Modell einfache oder sogar komplexe Systeme simulieren können. Eine weitere Möglichkeit sind Testbeds, die sich insbesondere während der Entwicklung eignen, um unter anderem Stresstests von besonders kritischen Funktionen durchzuführen, Sensorknoten im Netzwerk zu testen oder schnelle Effizienzanalysen unterschiedlicher Implementierungen durchzuführen.

Sowohl Testbeds als auch Simulationen haben jedoch einen entscheidenden Nachteil. Sie können das Verhalten einer Anwendung unter nicht deterministischen realen Bedingungen oder Umwelteinflüssen nur sehr begrenzt simulieren. Die Fehlersuche bei einem abschließenden Test unter realen Bedingungen gestaltet sich daher oft schwierig. Hier stehen nur wenig Informationen und Möglichkeiten zum Debuggen zur Verfügung und auch der Zugriff auf die Sensorknoten ist begrenzt.

Die in [3] vorgestellte Cross-Plattform (CP) unterstützt für solche Szenarien den Softwareentwicklungsprozess durch ein eingebettetes Test-Framework, dessen erster Entwurf in diesem Paper vorgestellt wird. Es ermöglicht Integrations- und Systemtests zur Laufzeit auf einer Zielplattform auszuführen, Peripherie oder Ereignisse zu emulieren, Fehler zu injizieren und Ergebnisse für die spätere Auswertung zu speichern.

II. KONZEPT

Änderungen durch ein Testsystem sollten so wenig wie möglich Einfluss auf die eigentliche Anwendung haben. Auch Anpassungen an den einzelnen Betriebssystemen (OS) sollten vermieden werden. Eine Anwendung auf Basis der CP greift

über OS-Wrapper auf OS-Funktionen zu. Sowohl Ereignisse die vom System erzeugt, als auch Funktionsaufrufe die von der Applikation ausgeführt werden, durchlaufen dabei die OS-Wrapper. Daher eignet sich diese Schicht um ein anwendungs- und plattformunabhängiges Testsystem zu implementieren. Grundlegende Funktionen wie das Speichern von Daten, virtualisierte Timer und ein integriertes drahtloses Code-Update stehen zur Verfügung.

A. Datenmanipulation

Die Architektur der CP wird genutzt um Aufrufe und Parameter in den OS-Wrappern bzw. Callback-Funktionen zu manipulieren. Dazu werden sogenannte Saboteure für einzelne Funktionen definiert und Funktionsparameter oder Rückgabewerte vor der eigentlichen Ausführung zur Laufzeit abgefangen. Diese beeinflussen gezielt das Verhalten vor der weiteren Verarbeitung oder ersetzen sogar ganze Funktionen. Die Ausführung eines Saboteurs kann durch die Nutzung von vordefinierten Templates mit Bedingungen oder Systemereignissen verknüpft werden. Somit lassen sich Hard- und Softwarefehler sowie Ereignisse gezielt erzeugen und zum Beispiel für Paketverlustsimulationen oder Lasttests unter realen Bedingungen nutzen.

B. Aufbau

Für Tests werden die Sensorknoten in einem drahtlosen Testnetzwerk organisiert. Ein weiterer Sensorknoten vermittelt als Gateway zwischen diesem Testnetzwerk und einem Hostsystem. Zum Kontrollieren und Steuern der Testszenarios werden die Code-Update-Funktionen und Kommunikationsprotokolle der CP genutzt.

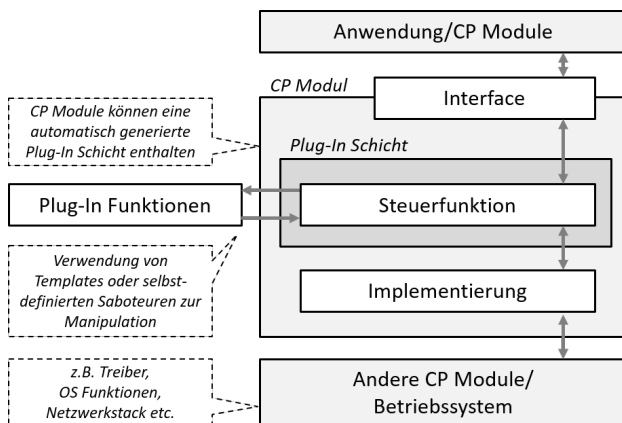


Abbildung 1. Pluginschicht in der Implementierung einer Funktion innerhalb eines Moduls in der CP zwischen OS-Wrapper und Anwendung

C. Testablauf

Das Test-Framework unterscheidet zwei Zustände, eine *Ruhephase* in der das System wie ursprünglich implementiert läuft und eine *Testphase* in der die Saboteure aktiv sind. Nach dem Systemstart befinden sich alle Sensorknoten solange in der Ruhephase, bis das Gateway den Test global startet. Ein

Testdurchlauf wird durch ein zuvor definiertes Zeitintervall begrenzt und nach dessen Ablauf automatisch angehalten. Diese Daten werden in einem externen Speicher abgelegt und mit Zusatzinformationen wie Zeitstempel und Infocodes zum Verursacher versehen. Somit kann der Fehler oder zumindest die Stelle an der er aufgetreten ist identifiziert werden. Am Ende jedes Testzyklus werden die Daten zur Auswertung über das Gateway zurück zum Hostsystem gesendet.

Die Implementierung eines Tests wird, im weitesten Sinne analog zu Hooks aus der Softwareentwicklung, durch eine Pluginschicht realisiert. Diese kann zur Compile-Zeit aktiviert werden und steht dann für die zu testende Funktion zur Verfügung. Die Pluginschicht steuert zur Laufzeit den Programmfluss und wechselt je nach Zustand des Testsystems zwischen Originalfunktion und Pluginfunktion. In einer Pluginschicht können mehrere Pluginfunktionen implementiert werden, deren Ausführung durch eine Steuerfunktion kontrolliert wird. Die prinzipielle Manipulation eines solchen Aufrufs innerhalb eines CP-Moduls ist in Abbildung 1 dargestellt.

D. Emulation

Eine begrenzte Hardwareemulation wird durch die Nutzung der OS-Wrapper innerhalb der CP möglich, um zum Beispiel Sensoren oder Bus-Kommunikation zu simulieren, falls diese zwar auf der Ziel-, nicht jedoch auf der Entwicklungsplattform zur Verfügung stehen. Ein Saboteur ruft dafür eine API-Funktion in der CP auf und erzeugt somit ein System-Ereignis.

III. AUSBLICK

Der Testablauf ist in der aktuellen Umsetzung durch Pointer-Manipulationen realisiert. Diese könnte in Zukunft vereinfacht und durch Makros zur Compile-Zeit festgelegt werden. Ein erneutes Aufspielen der Test-Anwendung wird vermeidbar, wenn ausschließlich die Testparameter vor jedem neuen Testlauf konfiguriert werden. Zudem ist eine automatisierte Stapelverarbeitung von Testszenarios mit unterschiedlichen Parametern im gesamten Netzwerk angedacht.

DANKSAGUNG

Diese Arbeit ist Teil des DIAMANT-Projekts am IHP und wurde durch das *Bundesministerium für Bildung und Forschung* unter der Referenznummer *03IPT601X* gefördert.

LITERATUR

- [1] K. A. Briski, P. Chitale, V. Hamilton, A. Pratt, B. Starr, J. Veroulis, and B. Villard, "Minimizing code defects to improve software quality and lower development costs.," tech. rep., IBM Rational Software, October 2008.
- [2] M. Ram and S. Kumar, "Analytical energy consumption model for mac protocols in wireless sensor networks," in *2014 International Conference on Signal Processing and Integrated Networks (SPIN)*, IEEE, February 2014.
- [3] M. Froberg, P. Poppe, N. Vetter, and M. Schölzel, "Cross-plattform zur hardware- und betriebssystemunabhängigen implementierung von anwendungen und protokollen," in *15. GI / ITG Fachgespräch Sensornetze*, pp. 47–48, Institut für Informatik und Computational Science, september 2016.

Moving Task Scheduling to Co-Processors in Energy-Harvesting Systems

Lars Hanschke
Research Group smartPORT
Hamburg University of Technology
lars.hanschke@tuhh.de

Alexander Sowarka
Hamburg University of Technology

Christian Renner
Research Group smartPORT
Hamburg University of Technology
christian.renner@tuhh.de

Abstract—New applications for Wireless Sensor Networks (WSNs) pose new demands on energy-harvesting systems due to increased complexity and power consumption. However, microcontrollers with high computing power are over-dimensioned for energy-aware activity adaption. An assistant processor which schedules activity and monitors the energy state of the harvester can increase the energy efficiency of the whole platform. Therefore, we use a sleek communication interface for exchanging task information between assistant processor and sensor node. Even with communication overhead, a factor of 35 in energy can be saved, which allows the whole platform to increase energy efficiency.

I. MOTIVATION

The popularity of Wireless Sensor Networks (WSNs), based on still increasing computational power, new sensing capabilities [1] or sharing of WSNs [2], [3], offers a plethora of new application scenarios. Sensor nodes are equipped with multiple sensors, e.g. fine dust, humidity or ozone, and different radio interfaces, e.g. LoRa [4], WiFi [5] or IEEE 802.15.4. With an increasing number of different peripherals, the complexity of the underlying program structure grows steadily.

Supplying sensor nodes with ambient energy from renewable resources, e.g. solar energy, allows for reducing the environmental footprint of WSNs and also decreases maintenance costs. In many scenarios, energy harvesting allows perpetual operation, but if the energy budget is restricted, e.g. due to physical size limitation of node and energy storage, the consumption of the sensor node has to be adjusted carefully.

While application scenarios, such as body-area-networks show potential for intermittently-powered devices [6], surveillance-related systems require continuous operation. Common techniques, e.g. as presented in [7] and [8], ensure Energy-Neutral Operation (ENO) by adjusting the energy consumption of the sensor node minding both energy intake and current energy level. However, adjusting the activity of a sensor node, requires knowledge about future energy intake. For solar energy harvesting, various approaches exist, e.g. [9] and [10], which all share the necessity for up-to-date information on the current energy intake to learn characteristics. The periodic wake-ups to sample the generated power at the solar panel adds an energy overhead to the system but is mandatory for size-restricted energy-harvesting systems.

While applications become more complex and sensor nodes more capable but also more power-hungry, the energy over-

head increases. Microcontrollers, which are designated for computing-intensive tasks, e.g. neural networks [11], operate very inefficiently for simple tasks, e.g. querying an ADC, due to overhead introduced by an operating system.

We argue that future sensor nodes powered by ambient energy need an assistant for energy purposes. This allows to easily exchange the microcontroller of the sensor node without adapting all energy-aware software. Additionally it allows for prolonged lifetime if power-hungry sensor nodes only concentrate on the designated task — rather than spending energy on adapting to harvesting intake. Approaches for adjusting the voltage of the node [12] or monitoring the energy-level [13] show the potential of assistant processors. However, a solution for energy-awareness including task scheduling is missing.

We show how an ultra-low-power microcontroller is used as a co-processor (COP) to assist the powerful but energy-expensive main processor (MP) in energy-aware behavior. Our approach encompasses the definition of a communication interface between both processors for task scheduling, computation of the energy-aware schedule for task execution and implementation of energy prediction schemes without the use of the MP.

II. DESIGN

To reduce the overhead of using the MP for energy-awareness, we introduce our design concept for an assisting COP. We shift scheduling of tasks, as well as energy prediction and adaption of energy consumption to the COP.

The basic principle is as follows: during an initialization phase, the MP transmits information about its tasks to the COP, which builds a schedule satisfying time and energy constraints. Following, the MP queries the COP about which task to execute and how long it has to sleep before waking up again. This is repeated after execution of each task.

First, we introduce the main architecture and give an overview of the used peripherals. Second, we explain needed software components and third highlight central aspects of the communication interface between both processors.

A. Architecture & Overview

Similarly to our approach in [14], we consider an energy-harvesting platform using a solar cell, a supercapacitor and

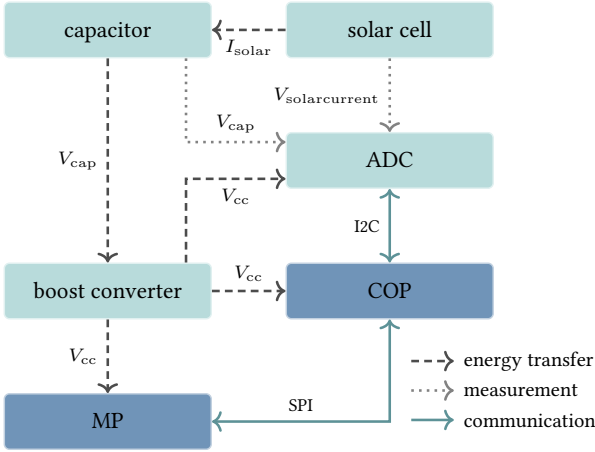


Fig. 1. Overview of the architecture; including communication between COP and MP as well as harvesting platform.

an external ADC. Figure 1 gives an overview of the platform. Incoming energy of the solar cell is directly fed into the capacitor. A small shunt resistor between solar cell and capacitor enables the ADC to measure the incoming energy of the solar panel. The energy level can be directly obtained from the voltage of the capacitor. However, our approach is not limited to this specific architecture as long as the COP has knowledge about the harvest intake and energy level of the sensor node.

The MP is configured by the developer to perform a distinct set of tasks. We assume a task to be an order of program commands, e.g. transmitting a packet, sampling a sensor or adjusting the air conditioning level. Since the application developer does not necessarily have knowledge about harvest prediction and adjustment of energy consumption, this responsibility is shifted to the COP.

To exchange information about tasks and scheduling, both processors are connected via a bus system of choice, e.g. I2C or SPI. Both follow the language of a communication interface, which we introduce in Section II-C.

B. Software Components

Our objective is to keep the software structure of the COP as flexible as possible; thus, we choose a layered software design, which is depicted in Fig. 2. Layers are only allowed to exchange information via defined interfaces.

In most microcontroller families, as the ARM Cortex M series [15], the manufacturer offers a Hardware Abstraction Layer (HAL) for interfacing with the microcontrollers and its peripherals. We rely on this library of STM, which allows for an easy exchange of COP hardware within the line of ARM Cortex M.

The COP implements four main components: ADC, communication, scheduler and prediction. Each component adheres to the layered design although not each component implements a functionality for each layer.

The ADC component handles interfacing with an external ADC. Methods for querying the voltage of the storage supercapacitor V_{cap} and the voltage at a shunt resistor $V_{solarcurrent}$ caused by the solar current, provide the information about the energy storage level and energy intake, respectively. The hardware layer ensures that the correct channel on the ADC is selected and the result is correctly converted to its physical representation, i.e. to millivolt and milliampere respectively.

The prediction component uses the energy readings to provide two important aspects: an estimation of the future energy intake and an energy budget ensuring depletion-free operation. Typically, prediction algorithms for energy harvesting, such as [9] and [10], give an estimate of the future energy intake within a fixed prediction horizon. This prediction horizon is additionally split into timeslots to decrease computational complexity. Our prediction component currently implements an Exponentially Weighted Moving Average (EWMA) filter with prediction horizon of 24 timeslots each spanning $T_{slot} = 1h$, as also used in [8]. Based on this estimate, algorithms adjust the activity of a sensor node by calculating the energy E , which can be taken from the capacitor without depletion. This energy $E = V_{cc} \cdot I_n^* \cdot T_{slot}$, with the assumption of a constant supply voltage V_{cc} is directly proportional to the average current I_n^* . In compliance with [8], I_n^* is called budget.

Based on the budget, the scheduler determines how many tasks the sensor node can fulfill within one timeslot. The scheduler models a task with a constant power $P = V_{cc} \cdot I_n$ which is consumed during the execution time t . In contrast to Dynamic Voltage Scaling (DVS) approaches like [12], this allows adjustment of the energy consumption only by altering the number of task execution. However, it is applicable without explicit changes in the supply voltage chain. Once the scheduler obtains the number of tasks to be executed, it evenly distributes the tasks within the timeslot to spread the current consumption. I_n and t per task have to be obtained from measurements by the developer and have to be transmitted during initialization phase of the communication between COP and MP, which we explain in the following.

C. Communication Interface

As mentioned in Section II-B, the communication between COP and MP is a very important aspect as it exchanges necessary information for task scheduling and energy budgeting. In general, two basic communication principles are possible: MP- or COP-initiated. While the latter allows the MP to shut off the Real-time Clock (RTC) circuit, it also requires wake-up capabilities on the communication bus, e.g. address matching. Since these vary between microprocessors, we opt for a MP-initiated communication interface.

1) *Task Definition:* To exchange information about the advised tasks, we implement a common task definition, which includes needed information for scheduling and energy consumption. Typically, the hardware configuration of a sensor node is known before deployment. Not necessarily all components have to be used simultaneously, but the capabilities of the node do not change over time.

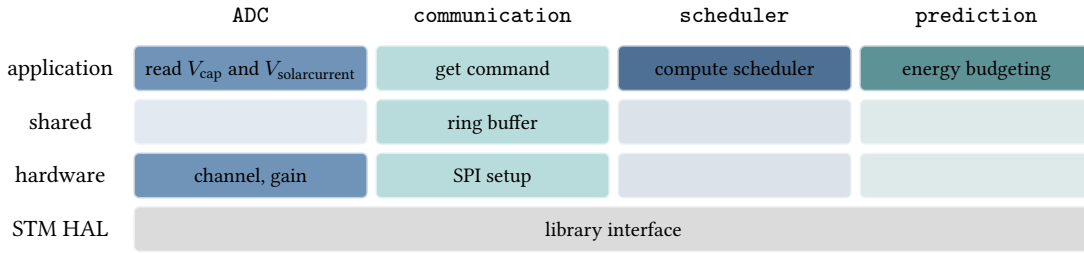


Fig. 2. Software layers of COP program; the hardware layer implements specialties of harvesting circuits, e.g. type of ADC or gain of the different channels; the shared layer contains information on the communication interface used by COP and MP; the application layer is independent of used hardware and easily exchangeable.

TABLE I
TASK PARAMETERS

name	description
state	μC -specific for power consumption, gives I_n
execTime	obtained at runtime or fixed
minInterval	time before execution is valuable again
maxInterval	maximum time after which execution is mandatory

This information is reflected in the *state* parameter for each task. An example for a task state is a turned-on radio interface and actively transmitting a packet. Right now, we assume that for each state, the developer has to measure the power consumption before node deployment. However, for future releases, we plan to measure this consumption online, e.g. by monitoring the decrease of capacitor voltage. Furthermore, the task definition contains time requirements: *execTime* reflects the time a sensor node needs to fulfill the advised task. It can be defined at compile time or measured at runtime by internal timers of the main processor. As Delay Tolerant Networking (DTN) [16] shows, tasks do not have to be executed as fast as possible — in most applications, a certain delay is tolerable. We reflect this matter by using a *minInterval* and *maxInterval* parameter for task definition. This allows the scheduler to effectively adjust the execution rate of task and thus adjustment of energy consumption. We summarize the parameters in Table I.

2) *Initialization*: Before the schedule is computed, the MP transmits the states and task definitions over the communication interface to the COP. At compile time, the COP does not know about the behavior of the MP. In this way, we ensure that a change of MP or the program structure of the MP does not affect the program of the COP. The initialization phase ends, when the MP may choose a prediction method, currently only EWMA, and forces the COP to compute the first schedule. If tasks change during runtime, e.g. due to remote commands [2], the MP is always capable of transmitting new tasks to its COP

3) *Regular*: At the end of each fulfilled task, the MP queries the COP about which task to execute next and how long to enter a sleep state before. The COP checks the list of next tasks, and answers with a previously announced unique task identifier and the sleep time before executing the task.

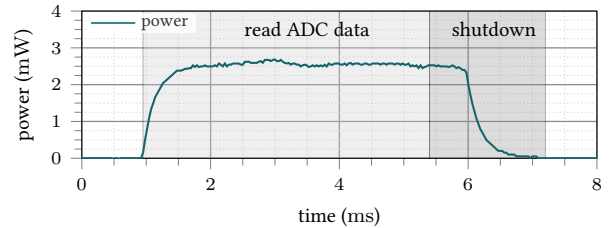


Fig. 3. Power consumption of the COP when querying the ADC of the platform; speed is limited by the conversion delay of the ADC.

The MP uses its internal timer to set the sleep time and, upon wake-up, executes the advised task.

III. RESULTS

The primary objective of the assistant COP is to reduce the energy cost for mandatory energy-aware measurements and calculations. This means that powering the COP and introducing the communication still has to consume less energy than powering the MP alone. Thus, we perform measurements of the different purposes of our platform and discuss the potential energy savings. We use the Espressif ESP32, with dual core Microcontroller Unit (MCU), up to 240 MHz clock frequency and integrated WiFi radio chip as the MP for our platform. The COP is a STM32L072RZ [15], which we run at 4.2 MHz. This allows to power down energy-costly clock circuits but still offers enough computing power. Both processors are supplied with $V_{cc} = 3.3 V$.

A. Measurements

We assess the power consumption of the COP by measuring the current at input side via an INA 139 measurement amplifier and record the time-varying current and supply voltage with the Keysight MSOX3014A oscilloscope.

1) *Energy-Awareness*: Due to the time overhead to boot the operating system, assessing the energy state of the platform is energy-intensive. We show the power consumption of querying the ADC in Fig. 3. Nearly constant 2.5 mW are consumed for 5 ms, while the execution is mainly limited by the communication with the ADC.

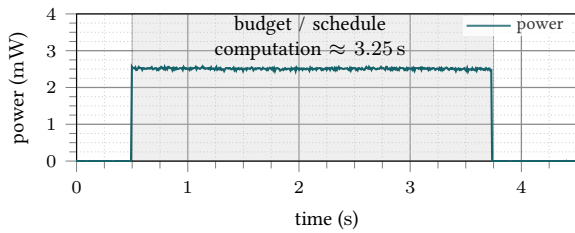


Fig. 4. Power consumption of the COP when calculating the energy budget for the next prediction horizon and calculate a schedule with four tasks.

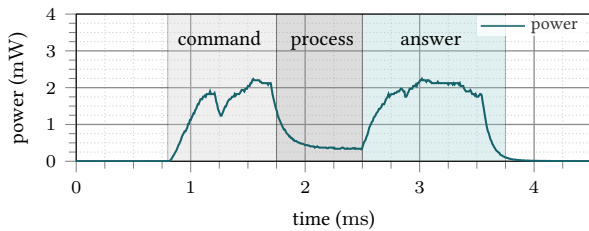


Fig. 5. Power consumption of the COP when communicating the advised sleep time and designated next task to the MP.

2) *Scheduling*: The energy level and harvest intake is used to perform the scheduling. This includes calculating the budget, but also distribute the advised tasks within the next timeslot. On the MP, this typically takes around 150 ms without booting the operating system. As shown in Fig. 4, the computation time increases by a factor of 22 to 3.25 s. However, the power demand of the ESP32 while computing is roughly 100 mW — which is a factor of 40 more.

3) *Communication*: In contrast to scheduling and energy-awareness, both MP and COP have to be powered for communication. To decrease energy consumption, this period should be as short as possible. Right now, we implement a synchronized and thus blocking communication on MP side but we plan to add an asynchronous communication interface. During runtime, the energy overhead for communication is dominated by the regular exchange of wake-up information, i.e. sleep time and designated next task. Figure 5 depicts the power consumption of the COP while communicating with the MP.

B. Discussion

Surely, the usage of an assistant COP for energy-aware scheduling adds costs, due to the additional microcontroller, and complexity, due to the need for a communication interface. However, it decreases the energy consumption of power-hungry and complex sensor nodes and thus increases efficiency, i.e. performing more tasks with the same energy. Assuming minutely wake-ups to gather energy state and hourly re-computation of budget and schedule consumes roughly 7% of the energy stored in a 100 F capacitor with the ESP32. In contrast, the combined approach with COP including regular communication with the MP only consumes 0.2% of the same capacitor — more than 35 times less.

IV. CONCLUSION

We showed the potential of an assistant processor for task scheduling for an energy-harvesting platform. Even with communication overhead, the energy savings due to monitoring of harvesting conditions, increase the overall platform energy efficiency. We plan to develop a harvesting platform for exchangeable MPs in future work to support spreading of energy-harvesting systems.

REFERENCES

- [1] C. Arora, N. Arora, A. Choudhary, and A. Sinha, “Intelligent Vehicular Monitoring System Integrated with Automated Remote Proctoring,” in *Intelligent Communication and Computational Technologies*. Springer, 2018, pp. 325–332.
- [2] J. Adkins, B. Campbell, B. Ghena, N. Jackson, P. Pannuto, and P. Dutta, “Energy Isolation Required for Multi-tenant Energy Harvesting Platforms,” in *Proceedings of the Fifth ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ser. ENSsys ’17. ACM, 2017, pp. 27–30.
- [3] T. La Porta, C. Petrioli, and D. Spenza, “Sensor-mission Assignment in Wireless Sensor Networks with Energy Harvesting,” in *2011 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, ser. SECON ’11. IEEE, 2011, pp. 413–421.
- [4] M. Cattani, C. A. Boano, and K. Römer, “An Experimental Evaluation of the Reliability of LoRa Long-Range Low-Power Wireless Communication,” *Journal of Sensor and Actuator Networks*, vol. 6, no. 2, 2017.
- [5] L. Hanschke, J. Heitmann, and C. Renner, “Challenges of WiFi-Enabled and Solar-Powered Sensors for Smart Ports,” in *Proceedings of the 4th ACM International Workshop on Energy Neutral Sensing Systems*, ser. ENSsys ’16. ACM, 2016.
- [6] J. Hester and J. Sorber, “The Future of Sensing is Batteryless, Intermittent, and Awesome,” in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys ’17. ACM, 2017, pp. 21:1–21:6.
- [7] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, “Power Management in Energy Harvesting Sensor Networks,” *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 6, no. 4, p. 32, 2007.
- [8] C. Renner, S. Unterschütz, V. Turau, and K. Römer, “Perpetual Data Collection with Energy-Harvesting Sensor Networks,” *Transactions on Sensor Networks (TOSN)*, vol. 11, no. 1, pp. 12:1–12:45, 2014.
- [9] C. Renner, “Solar Harvest Prediction Supported by Cloud Cover Forecasts,” in *Proceedings of the 1st ACM International Workshop on Energy Neutral Sensing Systems*, ser. ENSsys ’13. ACM, 2013.
- [10] A. Cammarano, C. Petrioli, and D. Spenza, “Pro-Energy: A Novel Energy Prediction Model for Solar and Wind Energy-harvesting Wireless Sensor Networks,” in *IEEE 9th International Conference on Mobile Adhoc and Sensor Systems*, ser. MASS ’12. IEEE, 2012, pp. 75–83.
- [11] S. Yao, Y. Zhao, A. Zhang, L. Su, and T. Abdelzaher, “DeepIoT: Compressing Deep Neural Network Structures for Sensing Systems with a Compressor-Critic Framework,” in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys ’17. ACM, pp. 4:1–4:14.
- [12] U. Kulau, D. Bräckelmann, F. Büsching, S. Schildt, and L. Wolf, “REAPer – Adaptive Micro-Source Energy-Harvester for Wireless Sensor Nodes,” in *Local Computer Networks Workshops (LCN Workshops), 2017 IEEE 42nd Conference on*. IEEE, 2017, pp. 1–8.
- [13] C.-W. Yau, T. T.-O. Kwok, and Y.-K. Kwok, “Energy Gatekeeper Architecture for Enabling Rapid Development of Energy-Harvesting Internet of Things,” in *Proceedings of the Fifth ACM International Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*, ser. ENSsys’17. ACM, 2017, pp. 43–45.
- [14] L. Hanschke, C. Renner, J. Brockmann, T. Hamann, J. Peschel, A. Schell, and A. Sowarka, “Light Insight — Emulation of Radiation Traces for Analysis and Evaluation of Solar-Harvesting Algorithms,” in *Proceedings of the 5th International Workshop on Energy Neutral Sensing Systems*, ser. ENSsys, Delft, Netherlands, November 2017.
- [15] *Datasheet STM32L072x8*, STMicroelectronics, 9 2017, rev. 4.
- [16] B. Gernert and L. Wolf, “PotatoScanner: Using a Field Sprayer As Mobile DTWSN Node,” in *Proceedings of the 12th Workshop on Challenged Networks*, ser. CHANTS ’17. ACM, 2017, pp. 47–49.

Bayesian Variational Optimization in Sensor Networks

Steffen Illium, Thomas Gabor, Thomy Phan

Mobile and Distributed Systems Group

LMU Munich

Munich, Germany

{steffen.illum, thomas.gabor, thomy.phan}@ifi.lmu.de

Abstract—This extended abstract focuses on the approximation of sampling times and spatial distribution of sensor networks and the application of variational Bayesian algorithms to optimize the deployment of such in an partially observable multi-unit setting. This work’s vision is to evolve networks of connected entities into an joint group that optimizes itself over time to become more energy efficient while sampling its environment at profitable time frames. Further more, communication expenses can be further reduced on the basis of a compressed data vector. Beginning with a problem statement, this work introduces methods in the context of distributed machine learning. Finally an optimization strategy, based on methods of Bayesian Variational Inference is presented.

Index Terms—sensory-networks, Bayesian Variational Inference, machine-learning, distributed learning

I. INTRODUCTION

The document at hand introduces an approach to overcome the problem of time and energy consuming sampling procedures while gathering vast amounts of environmental data, that are expensive to store and transmit for devices of the Internet-of-Things (IOT) level.

Sensor networks in general are a joint group of smaller devices, that are deployed to monitor their perceivable surrounding within an specific system. From bound application in factories or large machines (e.g. a ship) to environmental settings and biological observation tasks in nature, systems, applications and sensory equipments vary a lot. A sensor network in this work is therefore regarded as an interconnected group of multiple small hardware entities consisting of a SOC, FPGA, data storage and any sensory equipment to perceive their direct surrounding or more general discrete measurable events in time. Not less important is the availability of energy from a bound energy source like a built-in battery that constrains the available expenses to measure, compute and communicate. There also needs to be a kind of backbone-unit, that is capable of communicating with the group of entities. Furthermore, it should be able to perform large scale matrix calculations.

II. IOT-DEVICE LEVEL

With the increasing miniaturization and availability of small scale sensors, nowadays, close to two billion living human entities carry at least a single SOC powered networking devices in their everyday life, forming the biggest sensory network

ever known to mankind. Besides consumer level applications, which are already utilized by mobile operating system distributors (Google, e.g., consecutively improves its keyboard application), small task-specific sensor networks for scientific measures or commercial application are more accessible now than some years ago. While introducing longer standby times and enhanced computational efficiency, still, the problem of energy availability and total active periods when relying of an build in power source persists. Device-to-device or Device-to-Server communication still consume a large fraction of available power, especially when transmitting measurements wirelessly. [2] This problem is of particular significance when units are sparsely distributed or the availability of mobile networks is limited. Field-programmable gate arrays (FPGA), which are more or less single-purpose chips [7], make tasks such as visual analysis (e.g. object recognition) available on small-scale units, but increase the power consumption if applied to a lot of observations. This work will focus on optimizations that can be performed on spatial distribution of such units as well as sampling in relevant time-frames to keep energy costs low and therefore the system availability (w.r.t battery life and active periods) high. Especially w.r.t long-term monitoring tasks, this can be seen as a promising approach.

Since the efficiency of a distributed sensor network as described suffers from its main constraint, the availability of energy, a solution to utilize the available resources as best as possible needs to be found. By focusing on the optimization of sampling times and total unit distribution in a partially observable multi-entity setting, methods of Bayesian Variational Inference could help to approximate rewarding spatio-temporal states to reduce power expenses and total unit costs.

III. OPTIMIZATION BACKGROUND

This section will deal with the related background by covering the function of neural networks and Bayesian statistics in general.

A. Neural Networks & Multi-Entities

Recently, the field of machine learning gained more and more attention with the successful application of back-propagation in deep multi-layered neural networks. Compu-

tation by such is implemented within a graph-like structure by an activation function (e.g. *ReLU*, *Elu*, *Sigmoid*).

Neural networks in general are function approximators that learn to distort their input in a non-linear way to match visited training targets by adjusting their internal parameters. Such are called ‘deep’ (DNN) when involving multiple layers of computational distortion. [8] The *chain-rule of derivatives* (Eq. 1) enable small gradient corrections to be propagated through multiple layers of interconnected matrices of weights and biases.

$$\frac{\delta z}{\delta x} = \frac{\delta z}{\delta y} \cdot \frac{\delta y}{\delta x} \quad (1)$$

B. Bayesian statistic

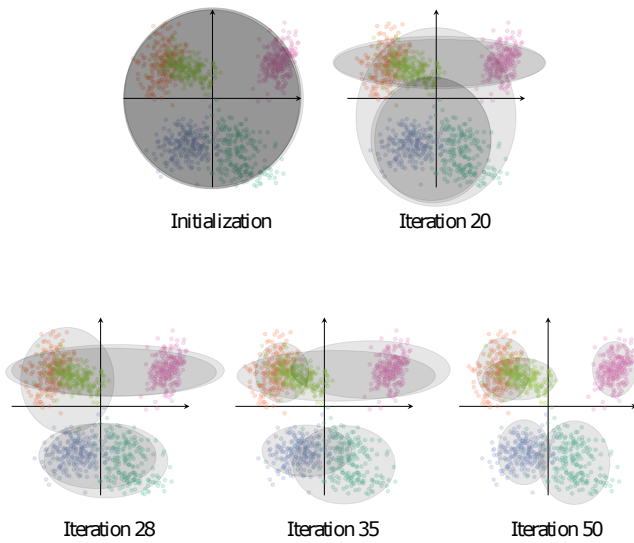


Fig. 1. Iterative parameter learning of a $k = 5$ Gaussian distributions [1]

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)} \quad (2)$$

Bayesian statistics allow the approximation of a distribution’s density function by calculating the *posterior* from the likelihood $p(x|z)$ (Eq. 2), given a series of observations x and a prior distribution $p(z)$. Small gradients are calculated by a distance function (often the Kullback-Leibler divergence (KL)) between prior density $p(x)$ and posterior density $p(z|x)$ given a latent variable z . The approximated density function $q(z)$ is shaped to approach the real density distribution by the alteration of z [1].

$$KL(q(z)||p(z|x)) = \int_z q(z) \log \frac{q(z)}{p(z|x)} \quad (3)$$

$$ELBO(q) = \mathbb{E}[\log p(z, x)] - \mathbb{E}[\log q(z)] \quad (4)$$

Since previously described modern neural networks can be utilized to approximate polinomial, continious and smooth

functions [9], Bayesian variational inference can be implemented and applied through such DNNs. *Kingma et al.* [6] introduced unsupervised auto-encoders that are capable of clustering observations $p(x)$ with a Gaussian constraint in a latent space, where z can be interpreted as a coordinate. Such networks learn a distribution’s parameters, for example variance and standard deviation of a single or multi-variant *Gaussian* [4]. A maximization of the evidence lower bound (ELBO) (Eq. 4) is often used in place of the KL divergence (Eq. 3)

In the following section, deep-learning by Bayesian density approximation will be related to the distribution of mobile sensor network entities.

IV. BAYESIAN OPTIMIZATION IN MULTI-ENTITY NETWORKS

Given a spatial and temporal natural phenomenon that needs to be monitored, networks of distributed units are often utilized to capture relevant observations on the broad variety of available sensors (e.g., seismic sensors for volcano activity monitoring). To gain sufficient information from such a multi-sensor installation, Bayesian optimization could be applied to approximate the most relevant sampling locations, as well as sampling time-frames for individual units or the group itself. On the other hand, the development of subgroups based on spatial densities (e.g., along a tectonic rift) should be possible. Time-dependent sampling could be foremost applied to cyclic events such as animal behavior (daily/sun-cycle) or on longer time-frames such as air-bound animal observation on certain days. Groups of sensory networks that observe major weather events in remote regions could be triggered by forecasting or simulation of system dynamics and then accordingly observed with the maximum available resources.

This could be achieved by the following procedure: Imagine a setting in which multiple distributed entities observe a phenomenon over a period of time on an evenly distributed spatial grid (if applicable) or rather a narrowed down distribution based on previous assumptions or a basic simulation. When initial measurements or previous observations and simulation of the system dynamics support the construction of a reasonably accurate prior $p(z)$, Bayesian variational inference seems to be a sufficient tool to achieve an optimization of the spatio-temporal observation distribution. The internal spatial prior would discribed by the units spatial distribution. The internal temporal prior on the other hand would then be a pre-set sampling frequency, tailored to the temporal dimension of the observation, as befor. At first, those units would have to learn to communicate their individual observations to a higher order backbone which correlates those. A sufficient way of observation vector compression should be applied to save energy in this step. [10] In a single unit setting, each unit could learn to compress environmental measurements accordingly, so that a backbone would be able to decode the information with a minimum loss. [5] Treated as multi-units in a partially observable setting, smaller entities could learn which information are sufficient to transfer along a small

communication vector. [3] This compressed common language between the sensory units and the backbone would minimize the communication cost so that the individuals each preserve energy while storing the raw measurement on a local storage space. Based on the simulation of observation, a single unit could learn how to communicate with the backbone unit. The learned compression function could then be implemented in the total group of sensors, or in a subgroup, to maintaining specialists, that focus on specific parts of their observation range.

After deployment, the backbone unit would then have to approximate temporal and spatial densities of the networks observations, to determine relevant spatio-temporal sampling sequences. Such a sequence is considered relevant when it is necessary to explain the target phenomenon. According to Bayesian optimization, this could be implemented by iteratively moving a latent prior distribution closer to the observed density distribution by the presented variational methods (see Figure 1). To match the spatial density distribution, units would need to be rearranged until the optimized distribution is met. Temporal adaption on the other hand could be realized by rather small parameter update over a small communication channel.

V. CONCLUSION

This work sketched the application of Bayesian Variational Inference to optimize the spatial distribution and sampling time-frames of a sensor network. Since relevant distributional parameters could be iteratively trained on a larger backbone unit, there is no need for training on IOT device level, so that SOC sensory networks with attached FPGAs could be solely used for observation purposes and feature compression before transmission. This would lead to longer standby-times on battery when observing natural events and therefore lower maintenance. On the other hand, more efficient spatial distribution would additionally reduces resource costs w.r.t unit count. As a consequence, the monitoring of spatially larger expanses would be possible at similar hardware expense. In addition, this methode could be used to determine optimal sampling points for spatially fixed installations.

REFERENCES

- [1] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.
- [2] C. Fischione. An introduction to wireless sensor networks. In *Swedish Communication Technologies Workshop (Swe-CTW 2014)*, 2014.
- [3] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 2137–2145, 2016.
- [4] A. Graves. Practical variational inference for neural networks. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2348–2356. Curran Associates, Inc., 2011.
- [5] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.
- [6] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.
- [7] G. Lacey, G. W. Taylor, and S. Areibi. Deep learning on fpgas: Past, present, and future. *arXiv preprint arXiv:1602.04283*, 2016.
- [8] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015.
- [9] H. W. Lin, M. Tegmark, and D. Rolnick. Why does deep and cheap learning work so well? *Journal of Statistical Physics*, 168(6):1223–1247, 2017.
- [10] D. Slepian and J. Wolf. Noiseless coding of correlated information sources. *IEEE Transactions on information Theory*, 19(4):471–480, 1973.

Extensions at Broadcast Growth Codes

Isabel Madeleine Runge

Department of Computer Science

University of Wuerzburg

Am Hubland, 97074 Wuerzburg, Germany

Email: runge@informatik.uni-wuerzburg.de

Reiner Kolla

Department of Computer Science

University of Wuerzburg

Am Hubland, 97074 Wuerzburg, Germany

Email: kolla@informatik.uni-wuerzburg.de

Abstract—In this article, Broadcast Growth Codes, a network coding based all-to-all data dissemination procedure, is adapted to the packet size and transmission time requirements of IEEE Std 802.15.4. In this context, packet overhead is reduced, constant packet sizes are replaced by dynamic packet sizes, and the codeword degree determination is extended by a stochastic estimation. Furthermore, the round-based approach was replaced by a time-based approach which uses non-synchronized nodes. In particular, the use of dynamic packet sizes improved the performance of the algorithm by approximately 30% in terms of latency.

I. INTRODUCTION

At the Internet of Things (IoT) or at Industry 4.0, we are often faced with a large amount of small, autonomous, cost-efficient mobile devices which are communicating with each other. Those devices can be sensor nodes which are collecting and forwarding sensor data.

Large-scale mobile networks with hundreds or thousands of sensor nodes pose many challenges concerning wireless communication. Here, collisions can occur, due to the big amount of transmissions, and cause high transmission loss rates. Unreliable data communication leads to reliable data maintenance being a key issue. If data has to be forwarded and gathered in one or few so-called sink nodes, the communication medium close to these nodes is highly stressed. Particularly in large systems, such bottlenecks are prone to traffic congestion and failures. Thus, bottlenecks should be avoided, for example by aggregating data in a distributed manner and by neglecting routing mechanisms. In this article, we are considering multi-hop communication, where each node aims to receive the originally collected data of each other node in the network. So there is not only one sink but all nodes act as source and sink nodes at the same time. Thus, bottlenecks and single point of failures are avoided.

A promising approach to reduce the amount of transmissions within a network is network coding, cf. [1]. At network coding, already received data is combined according to the chosen network coding procedure, and sent as a new packet to some or all neighboring nodes. Those neighbors can decode and recombine, or recode the received packets for further transmission. Consequently, routing is unnecessary and packets can reach their destination via random transmissions, eventually. The network coding approach which we discuss in this article is an advanced version of Multicast Growth Codes (MCGC) [2]. We adapted MCGC to the requirements of IEEE Std

802.15.4, which is a common standard for IoT applications. To differentiate between MCGC and the extension in this article, we will use the term Broadcast Growth Codes (BCGC) for the extended procedure in the following. The all-to-all data dissemination procedure presented in [2] uses XOR-combination of previously received data, and is based on LT Codes, cf. [3], and Growth Codes, cf. [4], [5].

To sum up, we are assuming a large-scale, distributed, mobile system, which uses wireless multi-hop communication and broadcast transmissions. Each node aims to receive the originally collected data of each other node in the network, and thus, bottlenecks and single point of failures are avoided. The considered optimization criteria are the number of necessary transmissions and the latency of the procedure.

Our main contributions are:

- We reduced the necessary BCGC packet overhead, and adopted the IEEE Std 802.15.4 at BCGC, including packet structure and size requirements. As a consequence, the transmission time of a packet can be considerably shorter and the algorithm terminates earlier.
- Besides that, we introduced dynamic packet sizes at BCGC, which can improve the performance of the algorithm in terms of latency by approximately 30%.
- Additionally, we extended the codeword degree determination by a stochastic estimation.

This article is structured as follows: After the necessary preliminaries, including IEEE Std 802.15.4 specifications, and a description of the system setup, the procedure and packet structure of BCGC and Forwarding will be described in Section II. In Section III, extensions of BCGC in comparison to MCGC are presented and illustrated by simulations. A conclusion follows in Section IV.

II. PRELIMINARIES

In this section, preliminaries like IEEE Std 802.15.4 specifications, the addressed system setup, as well as the procedure and packet structure of both BCGC and Forwarding are described.

A. IEEE Std 802.15.4 Packet Size Specifications

In IEEE Std 802.15.4, cf. [6], physical layer packets are restricted to a maximum of 133 Bytes, when considering 2.4 GHz. This includes an overhead of 6 Bytes and a Physical Payload (PHY Payload) of 0 – 127 Bytes as only 7 bits are used to determine the length of the PHY Payload.

Considering physical and MAC layer, a data packet has an overhead of 17 Bytes, and 116 Bytes remain for overhead and payload in upper OSI layers. Consequently, 116 Bytes are available for the BCGC frame, which will be discussed in Section II-C in more detail.

B. System Setup and Definitions

We consider a system of N homogeneous mobile sensor nodes with the ability to generate, store, and combine data. Data which is generated by a node will be called the node's own *data set*. We assume data is generated only once by each node and the data sets of all nodes have the same size.

Using wireless communication, a node is able to receive or transmit data packets. Each packet contains a so-called *codeword* as payload and a header with necessary overhead. More details on the structure of the header will follow in Section II-C. At BCGC, a codeword is created by a combination of data sets which have already been received in the considered node. This combination is done by XOR operations of the respective data sets. The number of single data sets being combined is referred to as the *degree* of this codeword.

At BCGC, a node is able to decode received codewords and stores the reconstructed original data sets. Therefore, it possesses a storage for reconstructed data sets and a so-called *waiting list* for received codewords which have not been completely decoded, yet.

C. BCGC Procedure and Packet Structure

The objective of BCGC is to be able to gather and reconstruct all N originally collected data sets of the N sensor nodes in each node. Before generating a codeword for transmission, a node determines this codeword's appropriate degree d . According to this degree, d data sets of the node's storage are strategically selected and combined for the new codeword. The codeword is subsequently sent, together with all necessary information for decoding in its header, as a new packet. The header of a packet contains the unique IDs of the data sets which are combined for the packet's codeword. Additionally, it contains the node's desired degree for the codeword which will be received next. The payload of a packet only consists of the codeword itself. Decoding in a node is started as soon as a codeword has been received. All data sets in the received codeword which have already been reconstructed in the node are removed from the codeword. This is done by combining the codeword with these data sets via XOR. If the remaining codeword contains more than one still unknown data set, the remaining codeword is stored in the node's waiting list. If the remaining codeword only contains exactly one data set which has not been received yet, the codeword is considered as decoded and the respective data set is stored in the node. A more detailed description of this procedure can be found in [2].

In contrast to the original algorithm in [2], we added an upper bound $maxDeg$ for the used codeword degree as well as a node's requested data set. On top of that, we are admitting dynamic packet sizes, thus, the packet degree has to be added

to the header. Details will be described in Section III but are already included in the following frame structure description.

Assuming 2-Byte IDs, a BCGC frame contains

- a field for the IDs of the data sets which are used for the packet's codeword ($\leq (maxDeg * 2)$ Bytes),
- a field for the desired degree of the node ($\lceil ld(maxDeg) \rceil$ bits),
- a field for the used packet degree ($\lceil ld(maxDeg) \rceil$ bits) if dynamic packet sizes are used,
- a field for the ID of the node's requested data set, which is only sent when the node still has to reconstruct only few data sets (0 or 2 Bytes), and
- the payload.

A BCGC packet consists of the physical and MAC layer overhead (17 Bytes), cf. Section II-A, and the BCGC frame, as described before. On physical layer, the Frame Length Field determines the length of the MAC frame in Bytes. As a consequence, the length of the MAC Payload has to be indicated in Bytes. Hence, the length of a BCGC frame also has to be rounded to Bytes.

D. Forwarding Procedure and Packet Structure

Forwarding, which is also denoted by No Coding, is a procedure where data is received by a node and forwarded to the neighbors without changing or encoding this data. One advantage of this procedure is a very small overhead. However, as many redundant packets will arrive if packets are sent randomly via multi-hop, many packets have to be received at the destination in order to receive all original data sets.

Assuming 2-Byte IDs, a Forwarding frame only contains:

- the ID of the (payload) data set (2 Bytes), and
- the payload.

As a consequence, a Forwarding packet consists of the physical and MAC layer overhead (17 Bytes), cf. Section II-A, and the Forwarding frame (2 Bytes + payload size).

III. EXTENSIONS AT BCGC

The following section describes the extensions we made at BCGC to fit the IEEE Std 802.15.4, and to improve the performance of the algorithm in terms of latency.

A. Reduction of Packet Size at Round-based BCGC

In the following, we refer to the IEEE Std 802.15.4 [6] and assume a radio frequency of 2.4 GHz and a data rate of 250 kbit/s. Thus, the transmission time of a packet ranges between $544 \mu s$ for the smallest possible packet of 17 Bytes (0 Bytes MAC payload) and $4256 \mu s = 4.256 ms$ for the largest possible packet of 133 Bytes (116 Bytes MAC payload).

In [2], time is divided into discrete rounds, where a round is defined as the time that is necessary for all nodes to consecutively send a packet via broadcast. For this purpose, a slotted MAC protocol with synchronized nodes is assumed. Each node has its own assigned time slot for transmission within each round. As a consequence, no collisions occur. The round-based results in [2] can be converted to seconds if the number of rounds is multiplied by the number of nodes in the

network and the transmission time of a packet. These results will be called the converted results in the following.

Referring to IEEE Std 802.15.4, 116 Bytes are available for the BCGC frame, cf. Section II-A. Depending on the payload size, the BCGC packet overhead has to be adapted to fit the maximum possible packet size. For this purpose, the degree of a codeword, which is the number of single data sets that are combined via XOR in the codeword, can be reduced to a degree $maxDeg$. By limiting the degree to $maxDeg$, the packet size becomes smaller and thus, the transmission time of a packet is shorter. As a consequence, the algorithm terminates earlier if $maxDeg$ is chosen appropriately, but more packets have to be received to reconstruct all original data sets in a node. The original MCGC [2] need $\lceil N/8 \rceil$ Bytes to indicate if a respective data set is combined in the codeword. By contrast, we only add the IDs of the data sets which are actually used for the codeword to the BCGC header. In the case of a constant packet size and a chosen degree of $maxDeg$, only $maxDeg \cdot 2$ Bytes are needed if we are considering 2-Byte IDs. The degree $maxDeg$ does not have to be the highest possible degree that still fits the IEEE Std 802.15.4 packet size restrictions but can be chosen arbitrarily low. A lower $maxDeg$ causes smaller packets and a shorter transmission time. The higher $maxDeg$ is chosen, however, the more different data sets are combined in a codeword packet, so the less completely redundant packets are received. Consequently, less packets have to be received by a node to reconstruct all original data sets.

Using an appropriate $maxDeg$ and the corresponding transmission time, the converted results can outperform the results produced with non-limited degree and ineligible transmission time. In general, the choice of $maxDeg$ also depends on the given payload size.

The achieved results can even be improved if each node explicitly requests one data set. This request is realized by adding the corresponding ID to the header when only few data sets have not been reconstructed in the considered node, yet. If a node receives packets with requested IDs, it includes all requested data sets which it has already reconstructed to its next codeword. As $maxDeg$ represents an upper bound for the used degree, a maximum of $maxDeg$ data sets can be included. By adding one additional ID to the BCGC header, packet size increases by 2 Bytes, but less packets have to be received in the end.

B. Time-based BCGC

In contrast to [2], we use non-synchronized nodes which perform CSMA/CA and collisions can occur due to the hidden node problem. This means, we waive the concept of rounds and directly produce time-based simulation results. As many nodes transmit their packets at the same time, where due to CSMA/CA only few transmissions cause collisions, the results of the non-synchronized approach outperform the converted results of the round-based approach. In the following, we use the term time-based BCGC for this procedure.

C. Time-based BCGC with Dynamic Packet Sizes

Assuming slotted, synchronized transmissions, packets have to have the same size for each node and the size does not change in the course of a simulation. In real testbeds, nodes can send packets of different sizes, so we replaced the constant packet size by a dynamic packet size and only add necessary overhead to the packet header. Consequently, packets are small in the beginning and grow according to the used degree for a codeword packet until they reach the degree $maxDeg$. The extension of BCGC by dynamic packet sizes increased the speed of the algorithm significantly, which is depicted in Section III-E.

D. Time-based BCGC with Delayed Increase of Degree

On top of the use of dynamic packet sizes, we also included the option to omit increasing the desired degree if it is not beneficial in the end. If the original algorithm suggests to increase the degree, a stochastic estimation is executed to finally decide if the degree should be hold or indeed be increased. Here, the expected maximum number of codeword packets is used which is necessary to receive to reconstruct all original data sets in the considered node. This expected maximum number depends on the number of data sets which have already been reconstructed in the node and on the degree of the codewords which will still be received. For the expected number of necessary packets, the formula for the coupon collector's problem is used for calculating the expected number E of degree-(1)-packets: $E = N \cdot \sum_{j=1}^{N-i} \left(\frac{1}{j}\right)$ for i already reconstructed data sets and N available data sets, i.e. $(N - i)$ data sets still have to be reconstructed. For degree-(d)-packets with $d > 1$, $1/p(Dist1)$ is used to determine the expected maximum number of degree-(d)-packets. Here, $p(Dist1)$ denotes the probability of a codewords to be immediately decodable after reception and thus, provide one additional reconstructed data set for the node. The higher the degree is, the less the probability is to receive a redundant packet, so the less packets have to be received. Further details have to be omitted due to restrictions in space.

The mentioned estimation multiplies the transmission time of a degree-($d+1$)-packet with the expected maximum number of degree-($d+1$)-packets which are still necessary to receive. Subsequently the results are compared with the respective results for degree-(d)-packets. If the applied stochastic estimation shows that the procedure would probably terminate earlier if the degree is hold, the degree will not be increased. Thus, the increase of the degree is delayed or bounded even if $maxDeg$ has not been reached, yet.

E. Simulation Results

The following plots are based on simulations with $N = 1024$ homogeneous nodes. Parameters are chosen according to the setting in [2] for better comparability. A square simulation area with a side length of 128 units of length and a maximum transmission range of 6.83 units of length is assumed. Nodes are using the Log-distance path loss model, cf. [7], to determine

the signal strength at a receiver. A node receives a packet successfully if the signal-to-noise ratio during the transmission has been high enough. Furthermore, a Random Walk Mobility Model is used to describe the nodes' movements.

Due to limitations in space, we only show simulation results for a payload size of 1 Byte and 20 Bytes, and a high degree of movement ($mo = 10$), where the number denotes the maximum permitted speed for a node in units of length per 100 *m.s.* Please note that ($mo = 10$) represents an upper bound of reasonable movement, as any further increase would not change the results significantly. Since we do not want to restrict to one specific application, we used the maximum possible speed in the depicted simulations.

The following graphs show simulations of different extensions of time-based BCGC where nodes perform CSMA/CA compared to simulations of Forwarding (yellow), which is described in Section II-D. The first time-based BCGC approach (green) is only extended by the use of an upper bound $maxDeg$ for the algorithm and the use of a node's requested data set, according to Section III-A and Section III-B. Depending on the respective payload size, the limit $maxDeg$ is selected so that the packet still fits the IEEE Std 802.15.4 requirements concerning packet size. The second BCGC approach (blue) is additionally extended by dynamic packet sizes, cf. Section III-C. The third BCGC approach (red) includes all extensions which are described in this article, in particular, the extension by the delayed increase of the desired degree in Section III-D. In comparison to the first approach, the final extension can improve the performance by approximately 30%, cf. Figure 1. The higher the payload size is, the lower $maxDeg$ has to be chosen to fit the IEEE Std 802.15.4 requirements. A reduced $maxDeg$ results in a smaller difference between constant and dynamic packet sizes. Nevertheless, the difference is still obvious, cf. Figure 2.

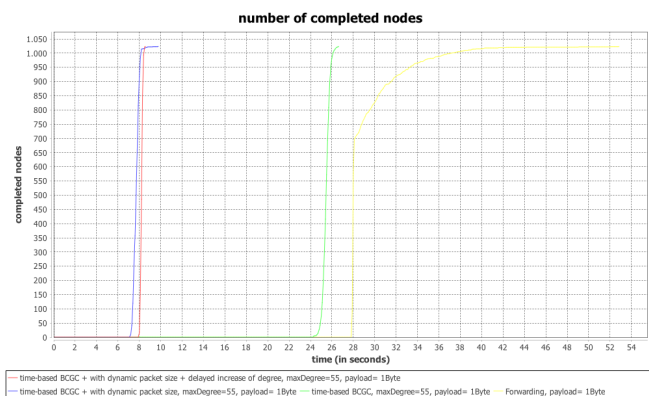


Fig. 1. Number of Completed Nodes 1Byte Payload

IV. SUMMARY AND CONCLUSION

To sum up, we extended Multicast Growth Codes [2] to adapt to the IEEE Std 802.15.4, use non-synchronized nodes which perform CSMA/CA, and improved the performance of the algorithm in terms of latency. To this end, we reduced

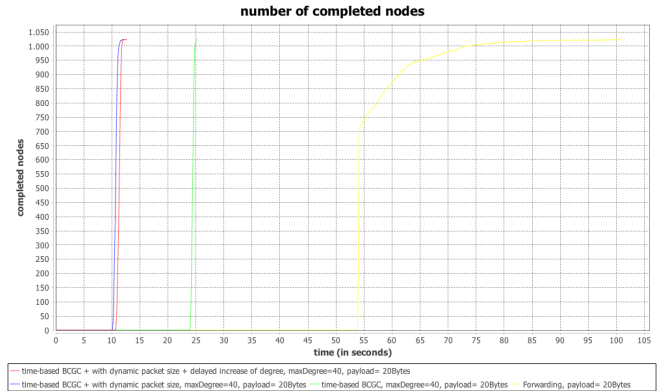


Fig. 2. Number of Completed Nodes 20Bytes Payload

the packet overhead by limiting the degree of transmitted codewords, instead of using the maximum possible degree N at a system with N different existing data sets. With this modification, more packets have to be received in a node to reconstruct all N original data sets but transmission times are considerably shorter. Shorter transmission times result in a lower probability for collisions, and the communication medium is less dense. Consequently, less packets have to be transmitted until a node receives a certain number of packets and is able to reconstruct all original data set.

On top of that, we replaced constant packet sizes by dynamic packet sizes and do not enlarge packets for transmission artificially. In particular, the use of dynamic packet sizes improved the performance of the algorithm by approximately 30% in terms of latency. Last but not least, we adapted a node's decision to increase the desired degree of a codeword within the algorithm. In [2], only the number of already reconstructed data sets was considered. We additionally consider the estimated time which will be necessary for the node to reconstruct all original data set if the increased degree and thus, the respective higher transmission times are used.

REFERENCES

- [1] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on information theory*, vol. 46, no. 4, pp. 1204–1216, 2000.
- [2] I. M. Runge and R. Kolla, "Mcgc: A network coding approach for reliable large-scale wireless networks," in *Proceedings of the First ACM International Workshop on the Engineering of Reliable, Robust, and Secure Embedded Wireless Sensing Systems*. ACM, 2017, pp. 16–23.
- [3] M. Luby, "Lt codes," in *Proceedings of the 43rd Symposium on Foundations of Computer Science*. IEEE Computer Society, 2002, p. 271.
- [4] A. Kamra, V. Misra, J. Feldman, and D. Rubenstein, "Growth codes: Maximizing sensor network data persistence," in *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 4. ACM, 2006, pp. 255–266.
- [5] A. Kamra, J. Feldman, V. Misra, and D. Rubenstein, "Data persistence for zero-configuration sensor networks," in *ACM Special Interest Group on Data Communications (SIGCOMM)*, 2006.
- [6] I. . W. Group, "Ieee standard for local and metropolitan area networks—part 15.4: Low-rate wireless personal area networks (lr-wpan)," 2011.
- [7] B. Dezfouli, M. Radi, S. A. Razak, T. Hwee-Pink, and K. A. Bakar, "Modeling low-power wireless communications," *Journal of Network and Computer Applications*, vol. 51, pp. 102–126, 2015.

Using High-Voltage Pulses for Opportunistic Data Transfers in Wireless Sensor Networks

Jana Huchtkoetter, Andreas Reinhardt

Department of Informatics

TU Clausthal, Germany

jana.huchtkoetter@tu-clausthal.de, reinhardt@ieee.org

Abstract—The deployment of embedded sensing devices on croplands and pastures paves the way for precision agriculture and farming applications. Changing conditions may, however, require the occasional reconfiguration of the resulting networks of wireless sensors, e.g., to modify data reporting rates or synchronize internal clocks. In [1], we have presented an opportunistic broadcast channel to forward such configuration messages to embedded sensing systems, and we revisit the principal insights in this paper. The transmitting station is realized by means of an electric fence energizer, a device frequently utilized in agricultural settings. On the receiver side, only little hardware additions to embedded sensing systems are required to capture the high-voltage pulses and decode transmitted configuration messages.

I. INTRODUCTION

The steady technological evolution has led to the availability of embedded wireless sensing systems for *smart farming* applications. By providing real-time updates on crop health and environmental conditions, such systems effectively support agriculturists in decision-making and have consequently established themselves as indispensable tools on many farms.

From a technological perspective, the employed sensor devices mostly rely on low-power hardware to ensure their long-lasting operation even when their energy budget is restricted. Reducing a platform’s demand for power is only possible to a certain extent, however. In particular, the on-board wireless transceiver often dominates the platform power consumption. To minimize its negative impact on the platform longevity, energy-efficient medium access control protocols have been proposed [2]. While their application often leads to a reduction of the energetic overhead for the collection of data from many sensor devices, they seldom provide a way to efficiently transfer control commands from a control center to sensor devices.

The use of secondary “wake-up” receivers, which activate the primary wireless transceiver only when it is needed [3], is one approach to increase energy efficiency. Wake-up receivers still introduce an energy burden for their operation, though, and generally require the primary radio device to be used for data communications. The alternative option chosen in [1] and revisited in this work is the use of a secondary radio channel with low data rate for unidirectional data communications. We exploit the almost ubiquitous presence of electric fences in farming scenarios to this end. In particular, we showcase how their periodic emission of high voltage pulses can be

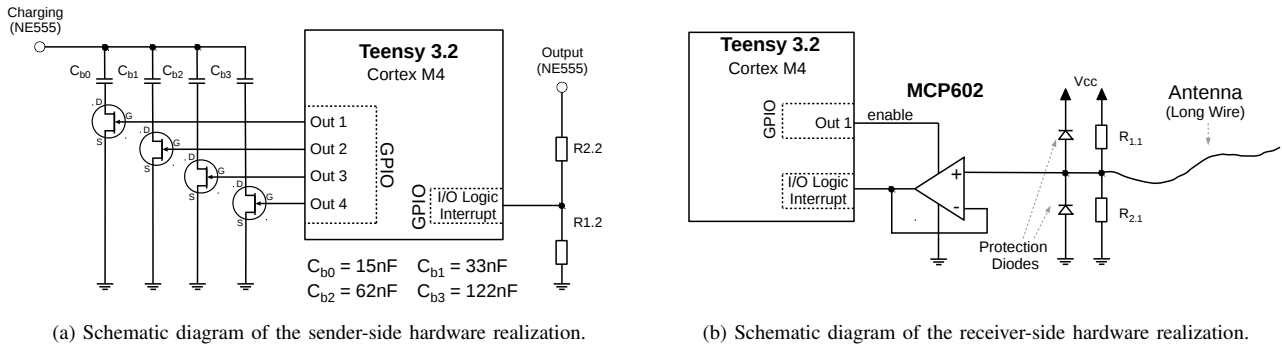
effectively used to broadcast control messages to all devices within its transmission range.

II. DATA TRANSMISSION VIA HIGH VOLTAGE PULSES

Electric fences are widely used in agriculture and farming. Placed around croplands, they prevent wild animals from trespassing onto arable land. Likewise, electric fences can be used to keep cattle within their pasturelands. Their operation is simple: Electric fence energizers periodically emit high voltage pulses onto fences made of electrically conductive material. Once an animal touches the fence, the resulting current flow is sufficiently painful to make the animal retreat.

The fact that fences usually fully encircle agricultural land combined with their high electrical conductivity makes them highly suitable candidates as transmission antennas for broadcasting configuration messages. Kulau et al. have demonstrated in [4] that the pulses emitted by electric fences can be detected by wireless sensor nodes. Building on this idea, we have conducted an in-depth study to find reliable ways to modulate data, devise antenna design considerations, and explore throughput limits. In this paper we highlight the most important aspects of the system design presented in [1], called PULSEHV, and showcase its operation.

To be able to send data over the electric fence, a way to encode information into the pulse train emitted by the fence energizer needs to be devised. In other words, a modulation scheme has to be chosen. During our applicability assessment of modulation schemes, we found the design choices to be limited. Regulations exist for electric fence energizers, such as IEC 60335-2-76 [5], which defines the maximum length of a pulse (10 ms), as well as the minimal time between two consecutive pulses (1 s). While no upper limit is defined to this interval, pulses need to be applied to the electric fence frequent enough to avoid animals trespassing it. A supplementary assessment of the shape of the pulse signal at different distances from the fence furthermore demonstrated that the only characteristic feature that could always be retrieved from the signal was the temporal displacement between successive pulses. We have consequently chosen to apply a pulse position modulation (PPM) to encode data within the intervals between emitted pulses. Our prototypical implementations of a corresponding sender and receiver design are presented in the following subsections.



(a) Schematic diagram of the sender-side hardware realization.

(b) Schematic diagram of the receiver-side hardware realization.

Figure 1. Schematic diagrams of sender and receiver peripheral devices to realize PULSEHV on the Teensy 3.2 microcontroller.

A. Sender Implementation

On the sender side, we have made the pulse interval variable through a minor modification of the electric fence energizer's timing circuit. The fence energizer used in our experiments (model Kemo FG025¹) internally relies on a NE555 timer, which determines its timings through two external resistors and one capacitor. Without modification, the fence energizer sends a pulse every 1.256 s. By integrating four switchable additional capacitors into its circuit, 16 timings for inter-pulse gaps between 1.256 s and 1.726 s can be configured. Thus, 4 bits can be transferred in each pulse gap at an average transmission rate of 2.7 $\frac{\text{bit}}{\text{s}}$. No further modifications to the electric fence energizer are necessary; only a microcontroller is needed to convert incoming data into 4 bit words and configure the capacitors accordingly.

We have decided to implement a minimum pulse position increment of around 30 ms between capacitor configurations, as this was shown to enable a reliable separation of pulse timings. Based on insights from practical experimentation, we have selected 15 nF as the minimum capacitance to add, and approximately doubled this value for each additional capacitor. In our prototype implementation, the concrete values measured using a calibrated RLC meter were 15 nF, 33 nF, 62 nF (nominally 63 nF), and 122 nF (nominally 120 nF). Through all combinations of the four capacitors, 16 different capacitance values could be realized.

At the same time, the primary functionality of the electric fence energizer, i.e., emitting regular electric pulses for farming applications, is not impacted. To increase the throughput further by encoding longer data words, the extension of the capacitor bank by a fifth value would have seemed meaningful. However, in order to allow for the correct distinction between pulses, the upper limit of the time between pulses is bounded by double the minimum pulse interval (i.e., 2.512 s in our case). Adding an additional capacitor, which would have to be double the capacity of the largest existing capacitor (i.e., 240 nF), would have violated this criterion. A schematic

visualization of the resulting sender-side circuitry, based on a Teensy 3.2 microcontroller, is shown in Fig. 1a.

The figure also shows the pulse feedback pin on the right-hand side. It signals an interrupt to the microcontroller whenever a high-voltage pulse has been emitted, and thus indicates the moment when changes to the capacitor configuration can be made without any negative impact on the timing accuracy. The set of active capacitors is determined through mapping the next four bits of input data to the switching transistors of the capacitor bank.

B. Receiver Implementation

In analogy to the sending side, we have designed the receiver devices based on the Teensy microcontroller. However, without loss of generality, any other low-power microcontroller device with accurate timestamping abilities can be used to receive data sent through the electric fence. The electric field is captured using a length of wire, which is connected to the input of an operational amplifier in voltage follower configuration. The large input impedance of an operational amplifier is required to minimize signal distortions. By connecting the operational amplifier's power supply to an output pin of the microcontroller, it can be selectively enabled when transmissions are expected to occur. Besides conserving power during the remaining time, this also allows for extended sleep cycles of the microcontroller which reacts to incoming pulses through interrupt handlers.

The choice of a long piece of wire instead of more sophisticated antenna options (e.g., a ferrite core) is based on the fact that electric fence energizers emit a purely electrical field. Without the presence of electromagnetic field components, inductive antenna components are inapplicable to receive the signals. We have supplementally run an experiment to compare the received signal strengths for these two different antenna types, and confirmed that the quickly changing potential between the fence and ground can be captured best using a length of wire. The modulated four-bit words are detected by the receiver as a combination of zero-crossings. These occur at a frequency of about 10 kHz and are followed by an offset final peak, as shown in Fig. 2. This specific combination of oscillation and final peak, coupled with the fixed time window

¹Specification available at <https://www.kemo-electronic.de/datasheets/fg025.pdf>

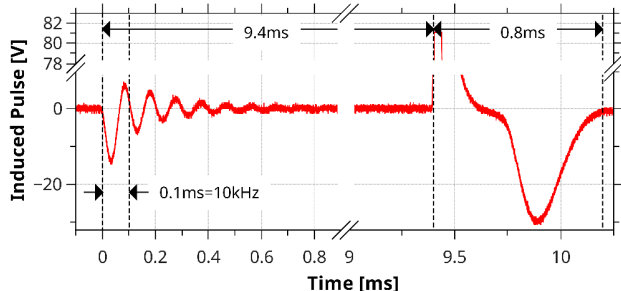


Figure 2. Signal waveform of the electric fence energizer, as received when using a length of wire as antenna.

for transmissions, allows the receiver to reliably detect pulses. The receiver implementation is shown in Fig. 1b.

III. EVALUATION

After having presented the design and implementation of PULSEHV for both the sender and the receiver devices, we show results from our evaluation of the interval timing stability and the throughput rates next.

A. Evaluation Setup

To evaluate PULSEHV, we have deployed a single sender device and a single receiver device (configured as shown in Fig. 1) in our laboratory. The distance between the sender's antenna and the receiver was approximately 1 m. The software implementation on the sender device modulates pre-determined data words through modifying the pulse positions, as described in Sec. II-A. On the receiving devices, a software implementation is executed to determine the length of the time intervals between the final peak of the preceding high-voltage pulse and the initial peak of the next pulse.

B. Consistency of Detected Pulse Intervals

To practically confirm the consistency of the pulse intervals of our PULSEHV sender design (cf. Sec. II-A), we performed a continuous transmission test over a period of more than 20 hours. The sender device has randomly selected a four-bit word for each transmission, and modulated it onto its emitted high-voltage pulse via PPM.

During the test, about 55,000 valid data words were detected, consisting of about 350,000 triggered events (cf. Sec. II-B). The rate of erroneous transmissions was 4.98 %.

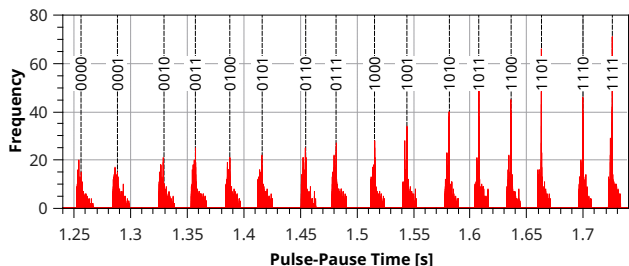


Figure 3. Histogram of experimentally measured pulse interval durations.

Table I
PULSE INTERVALS FOR ALL POSSIBLE COMBINATIONS OF THE ADDED CAPACITANCE, DETERMINED THROUGH PRACTICAL MEASUREMENTS.

Input	added capacitance	t_p [s]	$\min(t_p)$ [s]	$\max(t_p)$ [s]
0000	none	1.256 ± 0.0031	1.253	1.266
0001	15 nF	1.289 ± 0.0032	1.285	1.298
0010	33 nF	1.329 ± 0.0030	1.325	1.339
0011	48 nF	1.358 ± 0.0031	1.354	1.367
0100	62 nF	1.388 ± 0.0029	1.384	1.397
0101	77 nF	1.416 ± 0.0030	1.412	1.425
0110	95 nF	1.455 ± 0.0028	1.451	1.463
0111	110 nF	1.482 ± 0.0029	1.478	1.491
1000	122 nF	1.516 ± 0.0027	1.512	1.524
1001	137 nF	1.544 ± 0.0027	1.541	1.553
1010	155 nF	1.581 ± 0.0026	1.578	1.590
1011	170 nF	1.608 ± 0.0026	1.604	1.616
1100	184 nF	1.636 ± 0.0025	1.633	1.644
1101	199 nF	1.663 ± 0.0025	1.660	1.671
1110	217 nF	1.700 ± 0.0024	1.696	1.707
1111	232 nF	1.726 ± 0.0024	1.722	1.733

Fig. 3 shows a histogram of the observed pulse intervals, confirming that the chosen 30 ms pulse gap effectively prevents overlaps. A detailed analysis of the measured distributions of pulse intervals is provided in Table I.

C. Throughput Limits and Communication Errors

Due to the used modulation of pulse lengths, the time to transmit a word varies between 1.253 s (0000) and 1.733 s (1111). Assuming a uniform distribution of data words leads to an average data rate of one 4-bit word every 1.493 s. Thus, the achievable bandwidth for communication is $C = \frac{4 \text{ bit}}{1.493 \text{ s}} \approx 2.7 \frac{\text{bit}}{\text{s}}$.

As discussed in Sec. II-A, a fifth capacitor value to increase the pulse interval further cannot be included into the system design for practical reasons. Therefore, any messages larger than 4 bits in size need to be fragmented into multiple 4-bit words, and sent in a consecutive manner. Assuming a constant probability of incorrect pulse detections, however, it is expected that the probability of message transmission errors increases as the number of fragments per message grows. To identify whether dependencies exist between successive errors, we have analyzed the frequency and distribution of erroneous pulse detections in the collected log files of all received signals. Based on the data, it could be confirmed that erroneous pulse detections are independent of each other.

The properties of the time between incorrectly received words are summarized in Table II. While our analysis has shown that the median of the measured times between two erroneous pulse detections is 62.58 s, the inter-error interval even ranged from several minutes to over an hour during periods without errors. Besides these values, we could furthermore observe that most errors occur in bursts.

Table II
STATISTICAL ANALYSIS OF TIME BETWEEN ERRORS.

Min.	Lower quantile 25%	30%	Median	Upper quantile 60%	75%	Max.
0.13 s	1.54 s	5.26 s	62.58 s	183.61 s	398.83 s	4147.85 s

Such channel properties must be considered when choosing a data encoding scheme. In the mentioned test case, messages were only 72 bits long, resulting in just 18 transmissions and a simple parity calculation was sufficient to detect errors. Due to the unidirectional nature of the channel, the data were continuously repeated, such that missing fragments could be acquired in a later transmission. For larger messages, however, a more complex encoding scheme with the potential to detect (and possibly correct) errors immediately is recommended.

IV. CONCLUSIONS

We showcased a novel way for data transmissions, relying on the ubiquitously present fence energizers in smart farming scenarios. With only minimal modification to the hardware of the electric fence energizer, it can be turned into a unidirectional sender with a transfer rate of 20 bytes per minute. The receiver side only requires equally minor hardware modifications on systems with accurate timing and interrupt capability. The simple addition of an antenna and an operational amplifier enabled reliable reception of the pulses at the small cost of a minimal power increase. The opportunistic wireless channel could be shown to have a packet error rate of about 5%, of which most errors occurred in bursts.

ACKNOWLEDGMENTS

This research was supported by Deutsche Forschungsgemeinschaft (DFG) grant no. RE3857/2-1. The authors thank the Simulationswissenschaftliches Zentrum Clausthal-Göttingen (SWZ) for financial support, as well as Lars Wolf and Ulf Kulau for their scientific contributions.

REFERENCES

- [1] J. Huchtkoetter, A. Reinhardt, and U. Kulau, "PulseHV: Opportunistic Data Transmissions over High Voltage Pulses for Smart Farming Applications," in *Proceedings of the International Conference on Distributed Computing in Sensor Systems (DCOSS)*, 2018.
- [2] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, "The Evolution of MAC Protocols in Wireless Sensor Networks: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, 2013.
- [3] H. Ba, I. Demirkol, and W. Heinzelman, "Passive Wake-up Radios: From Devices to Applications," *Ad hoc networks*, vol. 11, no. 8, 2013.
- [4] U. Kulau, S. Rottmann, and L. Wolf, "Demo: Brzzz – A Simplistic but Highly Useful Secondary Channel for WSNs," in *Proc. International Conference on Embedded Wireless Systems and Networks (EWSN)*, 2017.
- [5] International Electrotechnical Commission, "Household and Similar Electrical Appliances – Safety – Part 2-76: Particular Requirements for Electric Fence Energizers," 2002, International IEC Standard 60335-2-76.

A Virtual Reality Multi-Sensor 3d Reconstruction System

Markus Friedrich, Kyrill Schmid
Ludwig-Maximilians-University Munich
 Munich, Germany
 {markus.friedrich / kyrill.schmid}@ifi.lmu.de

Abstract—We propose the concept for a distributed multi-sensor 3d reconstruction system that includes a virtual reality (VR) scan visualization component. It fuses depth data from two different sources: A mobile time-of-flight (ToF) depth sensor and an image-based 3d reconstruction module that estimates depth based on camera images. The continuously evolving 3d data set is visualized on-the-fly in a specialized VR application which allows for interactive scanning scenarios. Currently, two sensors are considered but we plan to extend the system to support a multitude of depth sensors that iteratively contribute to the global data set.

Index Terms—depth sensor fusion, distributed 3d reconstruction, virtual reality, data visualization

I. INTRODUCTION

The advent of cheap depth sensors shipped in consumer devices like the Microsoft[®] Kinect[™] or the Asus[®] Zenfone AR[™] has led to a boost in 3d reconstruction research (see e.g. [GPB11], [SKSC13], [FG11]) and 3d scanning application development. At the same time, virtual reality (VR) hardware has left the labs and is now mature enough for widespread adoption.

In this work, we introduce a concept that combines the best of both worlds through a distributed multi-sensor 3d scanning system that allows a user to scan real-world objects while providing instantaneous feedback on the quality and shape of the already measured object parts using a VR-based visualization.

The system uses a Zenfone AR[™] device which is equipped with a time-of-flight (ToF) sensor as main source for depth values. Since the used VR head-mounted display (HMD) comes with a front-facing camera, we would like to use it to compute additional depth values in complement to those provided by the Zenfone AR[™] using an image-based 3d reconstruction approach [SCD⁺06]. We exploit the fact that for each image, the pose of the camera is known since it is fixed on the HMD and the HMD computes its exact pose which is needed for VR applications. In addition, the internal camera parameters (e.g. focal length and pixel resolution) are well-known and do not change over time.

Our system consists of the following core components:

- A 3d reconstruction component that estimates depth values based on images that come from the front-facing camera within seconds ($\leq 2s$) for each incoming image.

- A depth value fusion strategy that merges depth values from different sources into a single consistent data set in near real-time (maximum delay $\leq 2s$).
- A VR visualization module that is able to blend depth measurements into the camera image of the real-world object without artefacts and noticeable delays in order to avoid VR sickness symptoms.

Currently, the system uses two depth sensors. Next steps involve the support of a multitude of depth sensors that all contribute to the global data set.

II. BACKGROUND

A. Rigid Body Poses

The 3d pose (position and orientation) of a rigid body (e.g. a camera) can be described by 4×4 matrices of the form

$$T = \left(\begin{array}{c|c} R & t \\ \hline 0_{1 \times 3} & 1 \end{array} \right),$$

where $t \in \mathbb{R}^3$ is the translation vector and R is a 3×3 rotation matrix. R is orthogonal and has a determinant of $+1$. T represents a rigid body pose and has thus six degrees of freedom (three for translation and three for rotation). The matrix T_{AB} represents the coordinate frame transformation from frame B to frame A .

B. Voxels & Truncated Signed Distance Functions

Voxels can be seen as 3d pixels that discretize a certain subset of the 3d real Euclidean point space. Each voxel stores a domain-specific n -tuple (e.g. signed surface distance and color). When stored in a regular grid, space complexity is $\mathcal{O}(n^3)$. More space efficient data structures exist, e.g. hierarchical space partitioning schemes or spatial hash maps [NZIS13].

A signed distance function $d : \mathbb{R}^3 \mapsto \mathbb{R}$ maps a point \mathbf{X} to a signed value that represents the distance between the point and the surface of an object. Thus, the surface is implicitly defined by the zero-set of d : $\{\mathbf{X} \in \mathbb{R}^3 : d(\mathbf{X}) = 0\}$.

Truncated signed distance functions (TSDFs) are only defined for a certain distance interval and are - in their discretized form - widely used for depth sensor fusion (e.g. in [GPB11]). Our system uses a voxel-based surface representation in combination with discretized TSDFs, where each voxel stores a TSDF value, a color and a fusion weight.

C. 3d Reconstruction

The goal of 3d reconstruction is to estimate the surface of real-world objects based on sensor data. Our system currently foresees two different depth value sources: A ToF sensor that measures the phase shift between an emitted and the corresponding reflected infrared light pulse in order to estimate the distance to an object. And a method that estimates depth using techniques from stereophotogrammetry that work on calibrated camera images [SCD⁺06]. The term calibrated means that for each image, the corresponding pose T_{WC} of the camera is known and can be used to transform a point from the camera coordinate frame to the world coordinate frame.

III. RELATED WORK

Real-time 3d reconstruction, depth value fusion and voxel visualization in real-time is a well-established field of research [GPB11], [SKSC13], [FG11]. One example is the work of Graber et al. [GPB11] that uses similar methods for 3d reconstruction and fusion than we do. The main difference to our work is the missing VR approach for visualization. A combination of 3d scanning and VR is proposed in [DSL15]. The authors describe a VR-based, user-guided 3d scanning simulator that uses the VR controller as a virtual depth sensor. Created point clouds are visualized within the VR application. While related to our system, it does not deal with real data.

IV. CONCEPT

The current system design foresees four hardware components: A workstation for sensor fusion and data set visualization (1), a workstation for 3d reconstruction based on camera images (2), a VR controller with mounted depth sensor (3, Asus[®] Zenfone AR[™]) and the VR HMD (4, HTC[®] Vive[™]) as display and camera.

The depth sensor is attached to the wireless VR controller in order to get its pose in the VR system's coordinate frame (up to a fixed offset). This is necessary in order to transform incoming depth values into the world coordinate frame for sensor fusion. The user holds this controller during the scan process and moves it around the object that should be scanned. The HMD displays the scene which contains both, the current camera image and the rendered global data set containing the already scanned object parts. Without showing the latest camera image, the user would lose orientation during the scan process. See Figure 1 for an overview of the system.

A. 3d Reconstruction

The 3d reconstruction component (Figure 1, component 2) uses a variant of the plane-sweep method to retrieve depth values from a series of calibrated images recorded by the front-facing camera [Col96]. It is optimized for modern GPUs and achieves near real-time performance on images with HD resolution.

Its principles are derived from basic triangulation. Corresponding image points are searched with help of pixel color-based similarity measures within calibrated views. With known correspondences in at least two views, the scene point can be

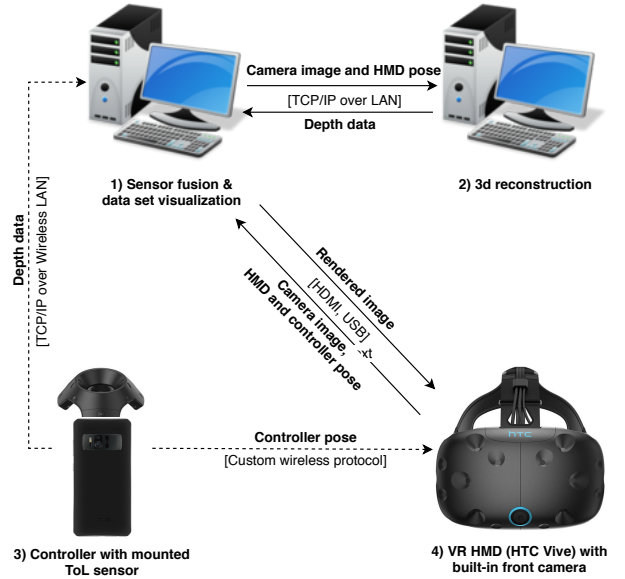


Fig. 1. Overall system architecture. Communication protocols are denoted in square brackets. Note that the selected VR system also uses two additional sensor boxes for HMD pose estimation. Since their role is transparent, they are omitted in this diagram.

estimated.

The detailed explanation of the method is based on Figure 2. A reference camera view and two other camera views with

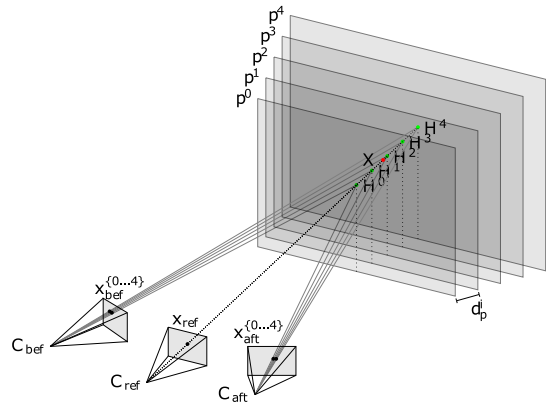


Fig. 2. The plane-sweep setting for a reference view, two sensor views and constant plane distances d_p^i .

centers C_{ref} , C_{bef} and C_{aft} with partially overlapping frustums are considered.

The scene point X (red point) corresponding to an image point x_{ref} (black point) in the reference view is to be estimated. X has to be located on the view ray v_{ref} through the camera center C_{ref} and the image point x_{ref} .

In order to estimate X , a set of n_p so-called sweeping planes $\{p^0, \dots, p^{n_p-1}\}$ parallel to the view plane of the reference camera are placed along v_{ref} . They divide the reconstruction volume. The distances between the sweeping planes, $d_p^i, i \in \{0, \dots, n_p - 1\}$, can be user-defined and constant or

determined by more sophisticated methods.

The intersection points between sweeping planes and view ray v_{ref} , \mathbf{H}^i (green points), are estimates of the location of \mathbf{X} . In order to determine the best candidate \mathbf{H}^b , all \mathbf{H}^i are consecutively projected on the view planes of the views V_{bef} and V_{aft} , resulting in image points $\mathbf{x}_{\text{bef}}^i$ and $\mathbf{x}_{\text{aft}}^i$ (black points). Correlation measures $c_{\text{bef}}^i, c_{\text{aft}}^i \in [0, 1]$ are then computed based on the color values of the view images I_{ref} and I_{bef} at $\mathbf{x}_{\text{ref}}^i$ and $\mathbf{x}_{\text{bef}}^i$ (resp. I_{ref} and I_{aft} at $\mathbf{x}_{\text{ref}}^i$ and $\mathbf{x}_{\text{aft}}^i$). Suitable correlation measures is e.g. the Euclidean distance. The best candidate index b is then determined by

$$b = \underset{i \in \{0, \dots, n_p - 1\}}{\operatorname{argmax}} (\max(c_{\text{bef}}^i, c_{\text{aft}}^i)).$$

Thus, the sweeping plane with the best overall correlation value is chosen. The plane-sweep method is parallelizable (one thread per image point in the reference view) and can therefore be executed efficiently on GPUs. In addition, the quality is adjustable by changing the sweeping plane distances d_p^i which alters the number of sweeping planes n_p .

B. Depth Sensor Fusion

All scene points estimated during a 3d reconstruction iteration and all new scene points from the ToF sensor need to be integrated into the global data set (Note that the ToF sensor automatically converts a scalar depth value to a full 3d scene point and all scene points are relative to the coordinate frame of their corresponding camera or sensor). This is done by the fusion component (Figure 1, component 1).

For fusion, a simple and well-known voxel fusion approach was selected [CL96]. For each voxel, the estimated signed distance D to the real surface is stored together with a fusion weight W which serves as a confidence measure.

Each scene point is transformed from camera-space to world-space using T_{WC} , resulting in the world-space scene point set $\{s_i\}$. The estimated distance to the surface, is defined for a certain interval t_{max} along the scene point's view ray v_i (with a distance of 0 at the exact position of the scene point). This represents the TSDF $d_i(\cdot)$ for that particular scene point defined on its view ray, see Figure 4 for an example.

The view ray interval in world-space is then discretized based on the edge length of a voxel e_v . For each sample point \mathbf{X} , $d_i(\cdot)$ is evaluated. The set of resulting signed distances is then fused with the voxels in the grid at \mathbf{X} . The new distance value D^* at grid position \mathbf{X} is computed by

$$D^*(\mathbf{X}) = \frac{W(\mathbf{X})D(\mathbf{X}) + w_i d_i(\mathbf{X})}{W(\mathbf{X}) + w_i}, \quad (1)$$

where w_i equals the correlation value of the corresponding scene point. The equation for calculating the new fusion weight W^* is

$$W^*(\mathbf{X}) = W(\mathbf{X}) + w_i. \quad (2)$$

Figure 3 shows an example of this fusion scheme.

In [CL96], a grid in combination with run-length encoding (RLE) for data compression is employed as voxel storage data structure. A more elaborate data structure based on hash

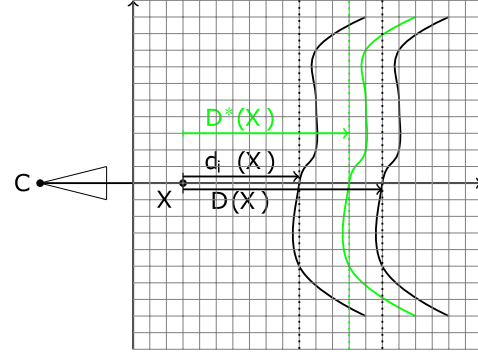


Fig. 3. The signed distance to the surface, $D(\mathbf{X})$, at position \mathbf{X} is updated by the signed distance, $d_i(\mathbf{X})$ at that position. The influence of both values is determined by the weights $W(\mathbf{X})$ and w_i . The new signed distance $D^*(\mathbf{X})$ is illustrated in green.

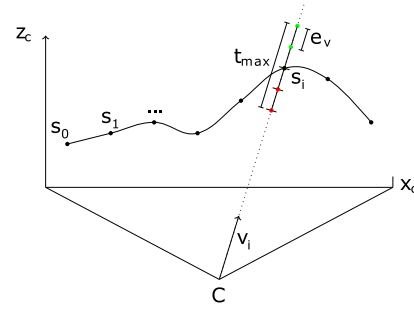


Fig. 4. Illustration of the TSDF value generation process reduced to the xz -plane in camera coordinates. Green dots indicate positive signed distances, red dots negative ones.

mapping principles was tested for the proposed system concept [NZIS13]. It stores small cubes of $8 \times 8 \times 8$ voxels in a spatial hash map with a hash function proposed by Teschner et al. [THM⁺03]. It reads

$$h(x, y, z) = ((xa) \oplus (yb) \oplus (zc)) \bmod n_b,$$

where $a = 86,028,157$, $b = 73,856,093$, $c = 19,349,663$ are large prime numbers, n_b is the number of hash map buckets and \oplus is the bitwise XOR operator. It is parametrized by the integer world-space position of the voxel cube $(x, y, z)^T$.

C. Visualization

The visualization of implicit surfaces stored in voxels can be done using sophisticated ray-tracers like the one employed in [Gra11]. Another option would be to generate a polygon mesh with help of the marching cubes algorithm [LC87] which is then rendered using standard 3d engines. Sophisticated 3d point cloud renderers can be employed as well but then only voxels with a signed distance of 0 can be considered in order to avoid cluttering.

We have implemented a basic renderer that retrieves voxel cubes from the hash map and renders small camera aligned rectangles for each voxel. See Figure 5 for details. Figure 6 shows a low-res voxel visualization with ca. 20,000 voxels.

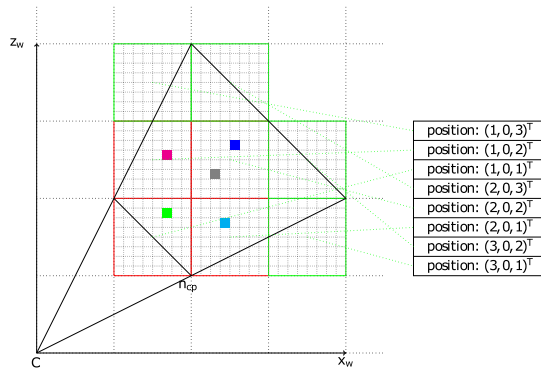


Fig. 5. The render pipeline. The scene camera frustum (black lines) is defined by the camera origin C , the near clipping plane n_{cp} and the far clipping plane f_{cp} (its orientation is indicated by the orientation of the black triangle). Eight potential voxel cubes (green and red lines) lie within or intersect with the frustum. Their positions are saved in the query voxel cube position list which is used to query for voxel cubes within the frustum. Note that only those voxel cubes surrounded by red lines are existent in the fusion data structure.

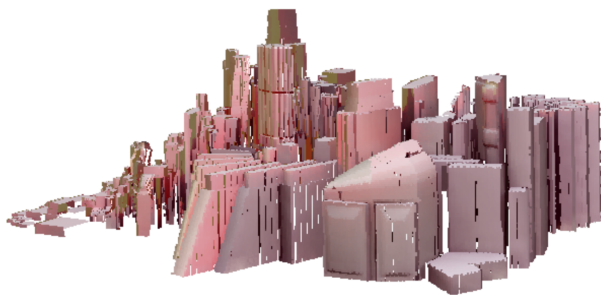


Fig. 6. Voxel set test rendering with around 195,000 voxels.

V. STATUS QUO, QUESTIONS AND FUTURE WORK

In context of this project, we prototyped several crucial parts of the system. We implemented the plane-sweep reconstruction method described in Chapter IV-A, the fusion technique and the hash map data structure discussed in Chapter IV-B for GPUs using OpenCL [Mun11]. Benchmarks considering a similar use case revealed the potential suitability of these techniques in terms of processing speed and robustness. More problem specific benchmarks are pending. We also experimented with different visualization techniques for implicit surfaces represented through discrete TSDFs. The method sketched in Chapter IV-C turned out to be a good trade-off between visual quality and speed. A first JSON-based message format to transport depth data from the mobile ToF sensor to the sensor fusion system was replaced by a binary format:

```
struct PointCloudMessage {
    uint msg_size, msg_timestamp;
    Matrix4x4f pose;
    Point4f[] points; // x, y, z, color
}
```

Listing 1. Binary point cloud transport message format.

The currently used sensor produces approx. 60,000 points/s which results in a bandwidth usage of approx. 900kb/s.

The following questions need to be addressed in future work:

- **Data distribution:** In future iterations of the system, multiple depth sensors should contribute to the scan result. The amount of sensor data might easily exceed the memory capacity of the GPU. Thus, intelligent data streaming and distribution schemes have to be developed.
- **3d reconstruction:** The plane-sweep method is vulnerable to complex real-world lighting conditions, especially if object surfaces have non-diffuse reflection properties. We need to evaluate more robust 3d reconstruction techniques.
- **Sensor fusion:** Each depth sensor type has its own maximum measurement precision. The fusion mechanism should account for that and prefer high-precision over low-precision measurements.
- **Visualization:** We need to investigate rendering techniques that are able to seamlessly visualize scan data together with real-world camera images.

Next steps include the integration of all system components into a coherent 3d reconstruction and visualization system.

REFERENCES

- [CL96] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on computer graphics and interactive techniques*, pages 303–312. ACM, 1996.
- [Col96] R.T. Collins. A space-sweep approach to true multi-image matching. In *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, pages 358–363, 1996.
- [DSLU15] Malvin Danhof, Tarek Schneider, Pascal Laube, and Georg Umlauf. A virtual-reality 3d-laser-scan simulation. In *Proceedings of the 2015 BW-CAR Symposium on Information and Communication Systems (SInCom)*, volume 2, 01 2015.
- [FG11] Simon Fuhrmann and Michael Goesele. Fusion of depth maps with multiple scales. In *ACM Transactions on Graphics (TOG)*, volume 30, page 148. ACM, 2011.
- [GPB11] Gottfried Graber, Thomas Pock, and Horst Bischof. Online 3D reconstruction using convex optimization. *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 708–711, November 2011.
- [Gra11] Gottfried Graber. Realtime 3D Reconstruction. Master’s thesis, Graz University of Technology, 2011.
- [LC87] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, August 1987.
- [Mun11] Aaftab Munshi. The OpenCL Specification Version: 1.1. <https://www.khronos.org/registry/cl/specs/opencl-1.1.pdf>, 2011. [Online; accessed 28/06/2018].
- [NZIS13] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3D Reconstruction at Scale using Voxel Hashing. *ACM Transactions on Graphics (TOG)*, 2013.
- [SCD⁺06] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1, CVPR '06*, pages 519–528, Washington, DC, USA, 2006. IEEE Computer Society.
- [SKSC13] F. Steinbruecker, C. Kerl, J. Sturm, and D. Cremers. Large-Scale Multi-Resolution Surface Reconstruction from RGB-D Sequences. Sydney, Australia, 2013.
- [THM⁺03] Matthias Teschner, Bruno Heidelberger, Matthias Müller, Danat Pomeranets, and Markus Gross. Optimized spatial hashing for collision detection of deformable objects. *Proceedings of the 18th International Workshop on Vision, Modeling and Visualization*, 2003.

Challenges and Opportunities of Wireless Underground Sensor Networks

Idrees Zaman

Sustainable Communication Networks
 University of Bremen, Germany
 iz@comnets.uni-bremen.de

Anna Förster

Sustainable Communication Networks
 University of Bremen, Germany
 anna.foerster@comnets.uni-bremen.de

Abstract—Wireless underground sensor networks (WUSNs) allow the sensor nodes to remain buried underground, and still communicate with the other sensor nodes to establish a wireless network. This property of WUSN enable numerous underground monitoring applications which are not possible with the traditional wireless sensor network (WSN). Our initial investigations showed that commodity sensor nodes and protocols cannot be used directly due to the unique behaviour of the underground channel. Therefore, a sensor node called MoleNet is especially developed for the WUSN experiments. Our experience with WUSN is then used for its deployment in several applications i.e. agriculture monitoring, underground mines monitoring, fish farm monitoring, train and intruding elephant notification, remote sensing of tsetse flies.

I. INTRODUCTION

The evolution of wireless sensor network (WSN) has extended their usage in different environments, i.e. agricultural monitoring, underground pipelines monitoring, infrastructure monitoring, intruder detection, playing field monitoring, soil ecology, etc. Unfortunately, traditional WSNs cannot operate in an underground environment due to the challenges presented by the underground medium. In [1], researchers have investigated that the communication in an underground environment is affected highly by the composition of soil particles, the amount of water held by these particles and the operating frequency of the sensor nodes. Therefore, new sensor nodes and protocols are required to work seamlessly in an underground environment.

Wireless underground sensor network (WUSN) is an extension to the traditional WSN, where the sensor nodes are buried in the subsoil region and they communicate wirelessly with each other through the soil. The concealment property of WUSN allows the users to prevent vandalism of the monitoring infrastructure, which is not possible in traditional WSNs or monitoring based on data loggers. Also, traditional monitoring based on data loggers restricts the coverage of the observing field as the extension of wired sensors over large fields increase deployment complexity. However, a WUSN allows to easily extend the monitoring of a field by adding new sensor nodes without disturbing the existing deployment for underground monitoring, which is very time consuming for an underground environment.

In this paper, we share our experiences from over three years

of development and deployment of WUSN and all application scenarios, which we have encountered so far.

II. CHALLENGES OF WUSN

The distinctive nature of the underground environment restricts the usage of commodity sensor nodes and protocols, as it introduces several challenges, i.e. communication range, lifetime, and topology for an underground deployed WSN.

A. Communication Range and Hardware Availability

Initially, commodity terrestrial sensor nodes operating at 2.4GHz are used for WUSNs [14]. However, no operable underground to underground (UG2UG) communication range is achieved, as the propagation of electromagnetic waves is highly affected by the medium of propagation. In [1], researchers investigated in simulations that 300MHz to 900MHz is the most suitable frequency band for UG2UG communication. Further lower frequencies result in increased antenna size that hinders the deployments for an underground environment. However, there are no off-the-shelf sensor nodes in this frequency band, which are designed specifically for WUSNs and the transceivers in this frequency band are only available for 433MHz and 868MHz frequencies. Therefore, we designed a sensor node called MoleNet [10] for WUSNs, which operates at 433MHz. The sensor node uses Arduino compliant Atmega1284p microcontroller, real time clock for maintaining and scheduling sampling and sending information events, 1Mbit external EEPROM to locally save the data from sensors and special interface for volumetric water content (VWC) sensors for underground soil monitoring. Figure 1 shows our custom build sensor node MoleNet. A reliable UG2UG communication range of 12.1m at a depth of 30cm was achieved in a soil comprising only sand particles using our MoleNet sensor node [13].

B. Interrupted Wireless Communication

The unique nature of underground environment presents new challenges for wireless communication which are not associated with terrestrial environment. Soil is a non-homogeneous medium and is composed of several kind of particles, i.e. sand, silt and clay. The solid particles in any soil constitute 50% of the soil volume and rest of the soil is made up of pores which are either filled with air or

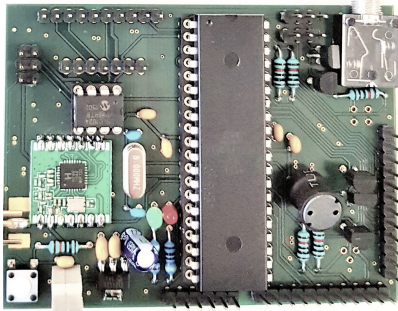


Fig. 1. MoleNet v3.0 based on Arduino compliant Atmega1284p micro-controller and RFM69 transceiver operating at 433MHz

water. The proportion of different particles in the soil and the amount of volumetric water content (VWC) in the pores affect the UG2UG communication [1]. Size of these pores also determine the water holding capacity of any soil. The water holding capacity is the amount of remaining VWC in any soil after two to three days of a rainfall, when most of the water is drained by the gravity. The water holding capacity of a sandy soil and clay soil can reach upto 25% VWC and 55% VWC respectively [18]. In [1], researchers investigated that a VWC of greater than 25% will result in no UG2UG communication. Therefore, in case of a rainfall the UG2UG communication can remain interrupted for several days depending on the type of the soil. These extended periods of communication interruption must be taken into account while designing any new protocols for WUSNs.

C. Lifetime and energy harvesting

Lifetime is one of the most critical factors in WSN, and it becomes even more important in WUSN as replacing the batteries after short durations is impractical in a WUSN deployment [12]. Also, energy harvesting methods available for above ground WSNs cannot be used for an underground environment, i.e. solar panel. Therefore it is required that the deployed WUSN should survive for several years. The interruption of communication also gives the opportunity to increase the lifetime of the WUSN by not opting to communicate in such scenarios and transmit only when the VWC is below a certain threshold. Hence, the WUSN requires extremely energy efficient protocols for data management on board and wireless communication in different scenarios to increase the lifetime. We are currently investigating several energy harvesting techniques for underground environment, i.e. pyroelectric and piezoelectric based methods for WUSN deployments, which would help in extending the lifetime of a WUSN deployment.

D. Topology Design

The deployment topology of the WUSN depends highly on the application and the environmental conditions around the observation place. Burying of the sensor nodes at different depths in the soil results in a three dimensional deployment

in WUSN, leading to different constraints for different communication links and different topologies. There can be three possible topologies for WUSN deployment, i.e. underground topology with same burying depth, underground topology with variable burying depths and hybrid topology which includes some above the ground relaying sensor nodes to increase the monitoring area of any field. An appropriate topology design should be considered depending on the monitoring application for enhanced power conservation and reliable wireless communication. Therefore, a WUSN testbed is deployed in the University of Bremen, to analyze different deployment topologies and the effect of environmental factors, i.e. temperature, volumetric water content (VWC) and composition of soil on different communication links [13]. Figure 2 shows the experimental field, that is especially created for agricultural and WUSN experiments in the University of Bremen.

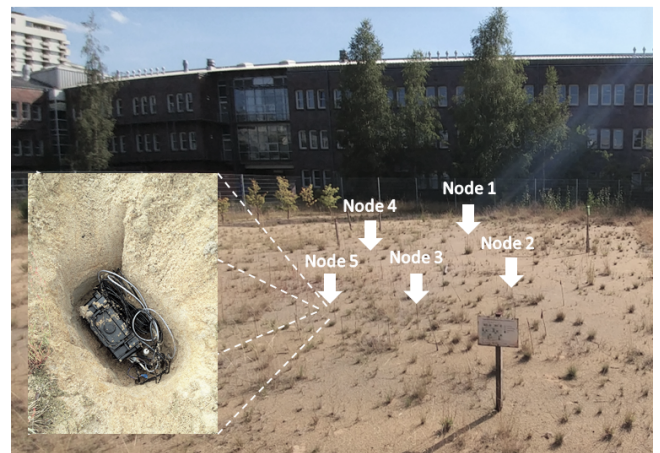


Fig. 2. Experimental field for WUSN in University of Bremen, Germany

III. APPLICATIONS OF WUSN

In recent years, the evolution in WUSN resulted in numerous applications. The concealment property of WUSN and its good wireless penetration through soil allow the extension of WSN in various applications, i.e. agriculture monitoring, underground infrastructure monitoring, underground mines monitoring, sports field monitoring. This section shows our experience with several applications using WUSN.

A. Agricultural Monitoring

WSNs are widely used for the monitoring of agricultural fields and optimized usage of resources for farming, i.e. in [15], [16] WSNs are used for precision agriculture and monitoring of different crops. However the traditional WSNs hinder the deployment of sensors and sensor nodes due to continuous agricultural activities i.e. ploughing, irrigation, etc. WUSN overcomes the shortcomings of traditional WSN for agricultural monitoring by burying the sensor nodes underground and allowing the farmers to continue their

regular agricultural activities.

Similarly, for an application of precision agriculture the ecologists required to analyze the efficiency of a revitalization technique for reforestation in Cameroon called Revitec [9]. The efficiency monitoring of the proposed technique requires constant monitoring of the moisture of the experimental field. The Revitec approach is deployed in remote fields in Ngaoundere, Cameroon and hence required the monitoring setup to be completely hidden to avoid vandalism. We used WUSN based on MoleNet sensor nodes to analyse the effectiveness of the Revitec approach. The deployed WUSN monitors the VWC and temperature of the experimental field in Ngaoundere, Cameroon.

B. Underground Mines Monitoring

The experience from an underground environment is then extended to its potential usage in underground mines. Usually wired, through the earth (TTE) or WSN based communication infrastructure is used for monitoring underground mines. In case of an incident, i.e. rockfall or explosion, traditional means of communication break down, resulting in no communication with the miners working underground. Lack of communication with the trapped miners makes it difficult for the rescuers to reach out to them.

We performed several experiments in the mock mine at the University of Witwatersrand, South Africa to analyse the performance of the WUSN in underground mines, particularly in a disaster scenario. Different disaster scenarios are emulated in the mock mine and the performance is compared with the traditional WSNs operating at 2.4GHz and 868MHz. Figure 3 shows an emulated disaster scenario, where a sensor node is buried under the rockfall. The MoleNet sensor node operating at 433MHz proved to establish the communication with the trapped miner in all emulated disaster scenarios [5].

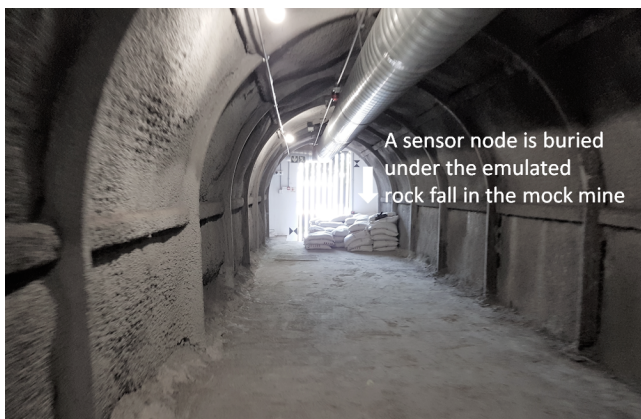


Fig. 3. An emulated disaster scenario in the mock mine at University of Witwatersrand, South Africa

C. Train and Intruding Elephant's Notification

In developing countries, the infrastructure of the railway does not always meet the safety standards. At many places, the railway crossings for the vehicles and humans are left unattended, which lead to dreadful incidents. Figure 4 shows an example scenario in Sri Lanka, where the railway crossing does not have any barrier. We proposed an advance notification system for the arrival of trains based on WUSN [17]. A sensor node equipped with geophone sensor is buried on the side of the railway track. The geophone (seismic sensor) can sense the vibrations of a train from 2-3km [2], and sends the notification to the sensor node at the railway crossing point. So without any existing infrastructure, the setup can be deployed anywhere resulting in a inexpensive solution for the notification of train arrival.

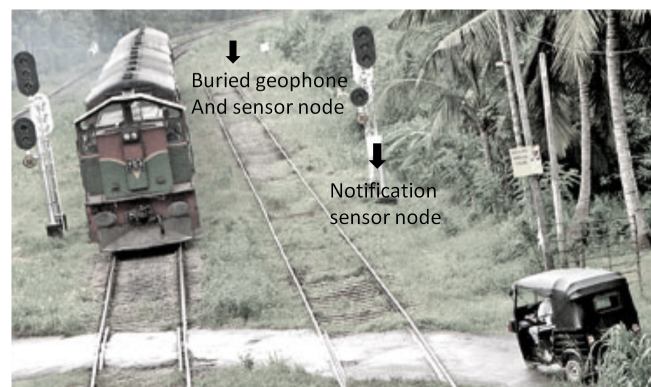


Fig. 4. Unattended railway crossing in Sri Lanka (sample picture) [11]

Similarly, in recent years, numerous agricultural fields and people in Sri Lanka are affected by the wild elephants [6]. Usually the elephants are scared off using electric fences, torches or firecrackers. However, electric fences are very expensive, inactive during frequent power outages and can be destroyed by the elephants. The same approach as for train notification can be used to notify the farmers about approaching elephants. The system can also be equipped with flash lights or firecrackers to disperse the herd of approaching elephants.

D. Fish Farm Monitoring

WSNs are also used for the monitoring of fish farms to optimize the usage of resources and increase fish production [7], [8]. The essential parameters for monitoring a fish farm include pH level, salinity, temperature, dissolved oxygen and other nutrients level. A similar research is pursued in the Mariculture Lab of Sam Nujoma Campus, University of Namibia, where the effect of different feeding regimes for fish is analysed in 12 different tanks. Recently, we replaced the traditional data logging system with a WSN based on MoleNet sensor nodes. Water monitoring sensors are attached with the MoleNet sensor node, that transfer the information to a central server. The system allows real time monitoring

of the fish tanks remotely. Figure 5 shows the pH, dissolved oxygen and temperature sensors connected to the MoleNet sensor node in one of the fish tanks. The usage of MoleNet in this environment is unique and novel: while the sensor nodes are not deployed under water (wireless communications would break immediately), the environment is extremely humid and obstructed. We have compared the usage of MoleNet to commodity sensor nodes and found MoleNet enables a more sparse and cost-efficient solution.

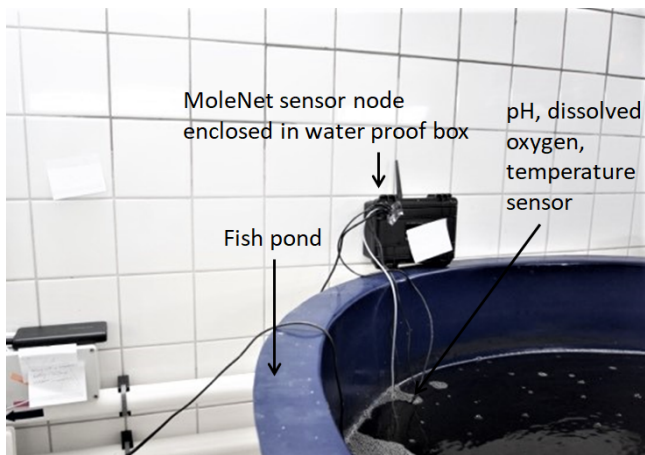


Fig. 5. Water monitoring sensors attached to the MoleNet sensor node at the Mariculture Lab in Sam Nujoma Campus, University of Namibia

E. Remote Sensing of Tsetse Fly

Tsetse flies belong to the blood sucking family of flies and are usually found in tropical Africa. Tsetse flies are the reason for transmitting trypanosomiasis (sleeping sickness) disease in humans as well as animals. Several techniques have been adopted for their localization and monitoring of their development and reproduction to eradicate trypanosomiasis [3] [4].

The female tsetse flies give birth to fully grown larvae, which rapidly become pupae in the soil. Together with epidemiologists, we explore the impact of soil properties on the development of the pupae in the soil with the ultimate target to disturb this development with natural interventions.

IV. CONCLUSION

We extended the use of WUSN in different environments where the traditional WSN cannot address the given problems. We showed that development of new sensor nodes and protocols is required for underground deployments. The positive results showed the potential of WUSN for diverse environments. If the mentioned research challenges in WUSN are addressed, then it can become the de facto standard for underground monitoring.

ACKNOWLEDGMENTS

We would like to thank our numerous partners for their support, time and effort. Most of the time, they have given us

the idea of how else to use our WUSN platform. We would like to thank Prof. Frederick Cawood from the University of Witwatersrand in South Africa; Prof. Hartmut Koehler from the University of Bremen, Germany; Prof. Dileeka Dias from the Moratuwa University in Sri Lanka; Dr. Thaddeus Gbem from the University from the Ahmadu Bello University in Nigeria; Dr. Jean Louis Fenji from the University of Ngauondere, Cameroon; and Laura Schnee from the University of Bremen. Many of the here described applications were developed by our students: Martin Gellhaar, Khushboo Qayyum, Muhammed Haseeb, Paul Meilahn, Arif Ahmed.

REFERENCES

- [1] Vuran, Mehmet C. and Akyildiz, Ian F., Channel Model and Analysis for Wireless Underground Sensor Networks in Soil Medium, Elsevier Science Publishers B. V, 2010, Phys. Commun, 245–254.
- [2] Chen, Q. and Li, L. and Chen, Li. and Peng, W., Seismic features of vibration induced by train, Acta Seismologica Sinica, 2004, 715–724.
- [3] Shereni, W. and Anderson, Neil E. and Nyakupinda, L. and Cecchi, G., Spatial distribution and trypanosome infection of tsetse flies in the sleeping sickness focus of zimbabwe in hurungwe district, Parasites & Vectors, 2016.
- [4] Gondwe, N. and Marcotty, T. and Vanwambeke, S. and Pus, C. and Mulumba, M. and Van den Bossche, P., Distribution and Density of Tsetse Flies (Glossinidae: Diptera) at the Game/People/Livestock Interface of the Nkhotakota Game Reserve Human Sleeping Sickness Focus in Malawi, EcoHealth, 2009.
- [5] Zaman, I. and Förster, A. and Mahmood, A. and Cawood, F., Finding Trapped Miners with Wireless Sensor Networks, 2018, Manuscript submitted for review.
- [6] Bandara, R. and Tisdell, C., Asian Elephants as Agricultural Pests: Damages, Economics of Control and Compensation in Sri Lanka, Natural Resources Journal, 2002.
- [7] H. F. Ragai, I. Adly, H. E. M. Sayour and S. Wilson, Remote control and monitoring of fish farms using wireless sensor networks, 2017 12th International Conference on Computer Engineering and Systems (ICCES)
- [8] Parra, L. and Sendra, S. and Garca, L. and Lloret, J., Design and Deployment of Low-Cost Sensors for Monitoring the Water Quality and Fish Behavior in Aquaculture Tanks during the Feeding Process, mmpi journal, Sensors, 2018
- [9] Koehler, H. and Heyser, W. and Kesel, R., REVITEC - An integrated ecological technology to combat degradation; first results from field experiments in Mallorca (Spain), Fourth International Conference on Land Degradation, Cartagena, Murcia, Spain, 2004.
- [10] Zaman, I. and Gellhaar, M. and Dede, J. and Koehler, H. and Förster, A., MoleNet: A New Sensor Node for Underground Monitoring, IEEE SenseApp, 2016.
- [11] Fazlulhaq, N., Protected or not, railway crossings are veritable death traps, Sunday Times, http://www.sundaytimes.lk/110612/News/nws_23.html
- [12] Stuntebeck, Erich P. and Akyildiz, Ian F., Wireless underground sensor networks: Research challenges, Adhoc Networks, 2006.
- [13] Zaman, I. and Haseeb, M. and Förster, A., Wireless Underground Sensor Network Testbed, 2018, Manuscript submitted for review
- [14] E. P. Stuntebeck and D. Pompili and T. Melodia, Wireless underground sensor networks using commodity terrestrial motes, 2006 2nd IEEE Workshop on Wireless Mesh Networks, Reston,
- [15] Z. Liao, S. Dai and C. Shen, Precision agriculture monitoring system based on wireless sensor networks, IET International Conference on Wireless Communications and Applications (ICWCA 2012)
- [16] L. Zhao, S. Yin, L. Liu, Z. Zhang and S. Wei, A Crop Monitoring System Based on Wireless Sensor Network, Procedia Environmental Sciences, 2011
- [17] Arif Ahmed, Advance Notification of Vehicles and Classification Based on Vibration Sensors (master's thesis), University of Bremen, 2018
- [18] Northeast Region Certified Crop Adviser (NRCCA) Study Resources, Soil and water management, Cornell University, 2010, <https://nrcca.cals.cornell.edu/>

RIZO - RFID basiertes Zeiterfassungssystem für Outdoorsportarten

Mario Redinger und Alexander von Bodisco
Hochschule für angewandte Wissenschaften Augsburg
Fakultät für Informatik
Augsburg

Email: mario.redinger1, alexander.vonbodisco@hs-augsburg.de

Zusammenfassung—Es gibt diverse Zeiterfassungssysteme [1] für Sport auf dem Markt. Aber da das Feld der Sportzeitmessung einen sehr eingeschränkten Kundenstamm hat, geht die Entwicklung nur langsam voran und einige Bereiche werden gar nicht bedient. Eines dieser Felder ist der Amateur Downhill Mountainbike Sport. In dieser Arbeit wird das Zeitmesssystem RIZO vorgestellt, welches eine preisgünstige Alternative zu etablierten Profilösungen darstellt und auf der neuen Funktechnologie LoRa sowie RFID basiert.

I. EINLEITUNG

Es existieren zwei gängige Verfahren auf dem Markt für diese Art der Zeiterfassung. Bei einem Verfahren trägt der Fahrer einen Smartsensor der beim Vorbeifahren an einem aktiven Sender die Zeit erfasst. Im anderen Verfahren ist am Rad/Fahrer ein aktiver Sender angebracht, mit diesem wird der Fahrer identifiziert sobald er durch eine Lichtschranke fährt. So kann seine Zeit vom Start und Ziel erfasst und ausgewertet werden.

Das RIZO Zeiterfassungssystem verwendet eine modifizierte Version des zweiten Verfahrens. Der große Unterschied hierbei ist, dass ein passiver und dadurch wartungsfreier UHF RFID (Ultra High Frequency Radio-Frequency Identification) Empfänger (Tag) genutzt werden kann, um den Fahrer zu identifizieren. Dies hat weiterhin den Vorteil dass der Sender leichter und wesentlich kostengünstiger ist. Eine weitere Neuerung zu den bekannten Systemen bietet die hier verwendete neue Funktechnologie LoRa. Bei anderen Systemen wird CB-Funk genutzt oder sogar auf die Funkübertragung verzichtet, so dass ein Kabel vom Start zum Ziel gelegt werden muss. Durch die Verwendung der LoRa Funktechnologie in Kombination mit passiven UHF RFID Empfänger wurde ein Zeiterfassungssystem realisiert, welches bei gleicher Leistung wesentlich kostengünstiger ist.

In Kapitel II werden zunächst die Anwendung und die daraus resultierenden Anforderungen diskutiert. Kapitel III gibt eine kurze Einführung in die eingesetzten Funktechnologien RFID und LoRa. Die für den Prototyp eingesetzte Hardware wird in Kapitel IV vorgestellt. Kapitel V beschreibt die Tests zur RFID-basierten Fahrererkennung/Fahreridentifikation und zur Datenübertragung via LoRa. Abschließend werden in Kapitel VI der aktuelle Stand des Systems und Erfahrungen mit der Technik wiedergegeben.

II. ANWENDUNGEN UND ANFORDERUNGEN

Die Anforderungen an das System hängen überwiegend von der Zielanwendung, der eingesetzten Hardware und den Gegebenheiten des Einsatzortes ab, weshalb bestimmte Eigenschaften unabdingbar sind. In diesem Projekt sind zunächst zwei Problemstellungen zu lösen. Die Fahrererkennung muss in der Lage sein mehrere Fahrer in kurzen Abständen auf einer bis zu 4 Meter breiten Strecke zuverlässig zu erkennen und zu identifizieren. Die Fahrer erreichen dabei Geschwindigkeiten im Zielbereich bis zu 50 Stundenkilometer. Im Ziel- sowie im Startbereich darf sich immer nur ein Fahrer im Erfassungsbereich des RFID Readers aufhalten, um eine eindeutige Zuordnung von Zeit und Fahrer zu gewährleisten. Je nach Fahrradgeometrie, Größe des Fahrers und Fahrsituation befindet sich der Kopf des Fahrers auf einer Höhe zwischen 130 cm und 200 cm. An dieser Stelle soll noch einmal auf die Zuverlässigkeit hingewiesen werden, da ein System von Endkunden nur bei einer hinreichend hohen Zuverlässigkeit von mehr als 99 % akzeptiert wird. Diese Zuverlässigkeit muss auch unter technisch herausfordernden Bedingungen, wie z.B. Nässe und Verschmutzungen auf Sender und Empfänger, erzielt werden.

Neben der Fahrererkennung stellt die Übertragung der Daten zu einem Webserver die zweite Herausforderung dar. Downhillparks erstrecken sich i.d.R. über mehrere Kilometer in voralpinem bzw. alpinem Gelände. Eine Sichtverbindung zwischen Start und Ziel ist auf Grund der Terraineigenschaften nicht gegeben. Auf Mobilfunk basierende Technologien stellen durch schwankende und oft schlechte Netzabdeckung im Zielgebiet keine praktikable Lösung dar. Weiterhin muss die verwendete Funktechnologie eine stabile und fehlerfreie Datenübertragung unter jeglichen Witterungsbedingungen, wie z.B. Regen, Nebel und Schnee, zur Verfügung stellen.

III. FUNKTECHNIK

Im Hinblick auf die in Kapitel II genannten Anforderungen haben sich die Funktechnologien RFID und LoRa herauskristallisiert. Beide Technologien erfüllen die jeweiligen Anforderungen und die zugehörige Hardware ist auf dem Markt zu günstigen Preisen verfügbar. Eine kurze Einführung in die verwendeten Technologien wird in den beiden folgenden Abschnitten gegeben.

A. RFID

RFID beschreibt eine Technologie, mit der es möglich ist, ICs(Integrated Circuits) kontaktlos auszulesen. Wird hierfür eine Frequenz höher als 860 MHz verwendet, spricht man von Ultra-Hochfrequenz (UHF) [11]. Bei einer geringen Frequenz spricht man von Langwelle (30-500 kHz) oder Kurzwelle (3-30 MHz). In diesen Frequenzbereichen ist das Auslesen von ICs nur im Zentimeter Bereich möglich.

Ein RFID System besteht aus drei Bauteilen: einem Lesegerät (Reader), einer Reader-Antenne und einem IC mit Antenne, der als Tag oder receiver (Empfänger) bezeichnet wird.

Der Reader sendet ein hochfrequentes elektromagnetisches Wechselfeld aus, welches vom Tag empfangen wird. Dabei wird eine Spannung induziert, die ausreicht, um dem Tag das zurücksenden der Daten zu ermöglichen. Dieser kann somit vollständig passiv betrieben werden.

Alternativ zu den passiven Tags gibt es semi-aktive Transponder die eine hohe Reichweite zwischen 6 und 100 Metern besitzen. In diesem Fall ist allerdings eine eigene Spannungsversorgung notwendig.

Zur Normierung der Tags hat sich der ISO 18000-6 C Standard, auch als Gen2 bezeichnet, durchgesetzt. Dieser legt unter anderem fest, dass die Tags global einsetzbar sein müssen, was bedeutet, dass sie auf allen UHF Frequenzbändern (860 - 960 MHz) funktionieren. Die Reader hingegen müssen mit der regional vorgeschriebenen Frequenz (865 - 868 MHz in Deutschland) senden. Der IC auf dem Tag hat einen Speicher von mindestens 64 Bit.

B. LoRa

LoRa ist eine proprietäre Chirp Spread Spektrum (CSS) Modulationstechnik [13], entwickelt von der Firma Semtech [4]. Das CSS besitzt eine hohe Modulationsbandbreite zwischen 125 KHz und 500 KHz. Die hohe Modulationsbandbreite erhöht die Robustheit gegen schmalbandige Störungen und erlaubt gleichzeitig eine leichtere Dekodierung im Vergleich zu einer Schmalbandübertragung [2].

Das Verfahren erlaubt eine Datenübertragung über mehr als 10 km bei einer verhältnismäßig kleinen Sendeleistung von 100 mW [12]. Dabei werden Übertragungsraten von 0,3 – 19,2 kbps erreicht.

Die Einstellungen, welche bei den LoRa festgelegt werden können, sind: Modulationsbandbreite, Spreizfaktor und Fehlerkorrektur. Das Ändern der Werte führt zu einer schnelleren Datenübertragungsrate bei geringer Reichweite oder zu einer hohen Reichweite bei einer langsamen Datenübertragungsrate [2].

Die Reichweite anhand der Einstellungen kann mit Hilfe eines Programms [6] der Firma Semtech berechnet werden, hier müssen die LoRa Einstellungen und die Modul Daten angegeben werden, damit das Link Budget in dB berechnet wird, so wie die Sendezeit und der Stromverbrauch.

Da ein öffentlicher Funkkanal verwendet wird (868 MHz), muss das deutsche Amateurfunkgesetz [3] beachtet werden. In Deutschland sind die Sendeleistung sowie die maximale

Auslastung des offenen Sendebereiches rechtlich geregelt. Um in Deutschland Daten per Funk im öffentlichen Raum zu übertragen, ist eine Frequenz aus dem Short Range Device (SRD)-Band zu verwenden [7]. Ebenfalls darf die zeitliche Nutzung im gewählten 868 MHz Band ein Prozent der verfügbaren Sendezeit nicht überschreiten [3].

IV. HARDWARE

In diesem Kapitel werden zunächst die verwendeten RFID und LoRa Module beschrieben. Weiterhin wird der Aufbau des Prototyps in einem eigenen Abschnitt behandelt.

A. RFID Reader

Der verwendete RFID Reader ist von der Marke Chafon Technology [10] und trägt die Modell Bezeichnung CF-RU5016-EU. Das Modell ist äußerst preisgünstig, da der Reader mit Antenne nur rund 150 € kostet. Es ist ein kombiniertes Gerät mit Antenne und Elektronik in einem wasserfesten Gehäuse. Dieser Reader wird primär für Parkplatz Zufahrtskontrollen verwendet.

B. LoRa Module

Zur Funkübertragung der Daten wurden zwei unterschiedliche LoRa Module der Marke EBYTE getestet, das E45-TTL-100 und das E45-TTL-1W mit jeweils einen Frequenzbereich von 862-893 MHz. Die Preise für diese LoRa Module belaufen sich auf rund 20 € für zwei E45-TTL-100 und auf rund 40 € für zwei E45-TTL-1W mit jeweils einer Antenne (3,5dBi Antennengewinn).

C. Prototyp

Der Prototyp für das Zeiterfassungssystem verwendet einen Raspberry Pi zur Steuerung des Systems. Die Zeiten werden mit Echtzeituhr Modulen konstant gehalten und zum Starten bzw. Stoppen der Zeit werden Reflektionslichtschranken eingesetzt.

Der Wasserschutz des Systems wird gewährleistet durch die Verwendung von HT DN 110 Abwasserrohren in Kombination mit Schraubmuffen für die Kabeleinführungen. Abbildung 1(a) zeigt den Prototypen. Die Module und Peripherien werden über eine selbst entwickelte Platine an den Raspberry Pi angeschlossen. Dieser ist fest im Gehäuse auf einer 3D-gedruckten Halteplatte verschraubt, wie in Abbildung 1(b) zu sehen.

Die Stromversorgung des Prototypen wird mit 12V Bleigelakkus und Stepdown- Wandlern realisiert. Zum Laden der Akkus wird eine 30 Watt Photovoltaikplatte verwendet. Die gefahrenen Zeiten können auf der Webseite rizotiming.de eingesehen werden und es ist auch möglich die Zeiten der Fahrer zu vergleichen und Ranglisten zu erstellen.

V. TEST

In diesem Kapitel werden drei Tests genauer beschrieben. Die ersten beiden dienen der Evaluierung der RFID Kommunikation und der damit verbundenen Positionierung und Ausrichtung der Antenne und Tags. Der dritte Test beschreibt die Kommunikation via LoRa in alpinem Terrain über mehrere



Abbildung 1: Prototyp des RIZO Zeitmesssystems

Kilometer und gibt einen guten Einblick in das Potential dieser Technologie.

A. RFID Erfassungsbereich

Im Vorfeld war nicht bekannt welchen Erfassungsbereich der RFID Reader zuverlässig abdeckt. Aus diesem Grund mussten Tests durchgeführt werden, die Aufschluss über die optimale Positionierung des Readers geben.

Der Testaufbau wurde in Form eines runden Gitternetzes, ausgehend von einem RFID Lesegerät mit einem Radius von 10 m, strahlenförmig mit einem Öffnungswinkel bis maximal 40° in 10 Grad-Schritten, angelegt. Das RFID-Lesegerät ist auf einer Höhe von 130 cm montiert. Der Tag wurde an einem Holzstab angebracht, ebenfalls auf einer Höhe von 130 cm für einen ersten Test und für einen zweiten Test auf einer Höhe von 200 cm. Die Messpunkte sind in Ein-Meter-Schritten vom Lesegerät entfernt, sowie in jeweils Zehn-Grad-Schritten versetzt bis maximal 40° . Die Messung wurde mit einem Serialmonitor-Programm erstellt, das auf einem Laptop ausgeführt wurde. Das Programm RealTerm [8] sendet auf Mausclick einen Query-Hex-String (0x04 00 01 DB 4B) an den RFID-Reader und empfängt eine Antwort im Hex-Format. Der Reader antwortet mit dem Vermerk „nicht gefunden“ oder mit der ID des Tags. Der Query-Befehl sorgt dafür, dass nur ein Lesebefehl getätigt wird. An jedem Messpunkt wird dreimal ein Query-Befehl gesendet.

Dies ergibt bei 5 Winkeln \cdot 2 Höhen \cdot 10 Distanzschritte = 100 Messpunkte zu je drei Messungen pro Punkt. Somit gibt es für den ersten Test 300 Messwerte. Die Messung wurde im Freien vorgenommen, um Reflexionen von Wänden zu vermeiden. Der Tag wurde immer in einer parallelen Linie zum Reader gelesen. Die Grafik 2 zeigt eine grafische Auswertung der Messergebnisse. Die Messwerte zeigen, dass bei einem Abstand größer als sieben Meter keine Messungen mehr möglich sind. Auch der Öffnungswinkel von -20° bis $+20^\circ$ bei Distanzen über zwei Meter sollte nicht überschritten werden. Die schlechten Messwerte auf einer Höhe von 130 cm sind ungewöhnlich und nur schwer zu erklären da sie nur im

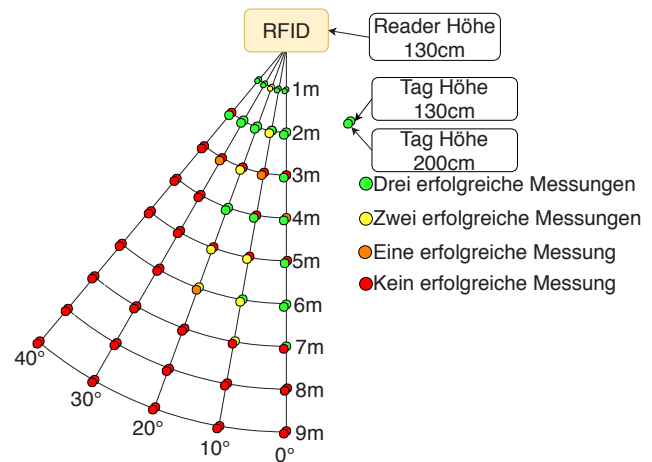


Abbildung 2: RFID Messergebnisse eines nicht bewegten Gen 2 passiv Tags

Bereich von 3 - 5 Metern nicht zufriedenstellend sind. Die guten Werte im Bereich von 6 - 7 Metern könnten durch Bodenreflexionen zustande gekommen sein. Die Ergebnisse für eine Messung auf 200 cm ist sehr zufriedenstellend, da es hier nur selten Übertragungsfehler gab. Die Schlussfolgerung aus diesen Werten ist, dass eine Höhendifferenz von ± 70 cm zwischen Tag und Reader gute Ergebnisse verspricht. Des Weiteren ist ein geringer Abstand von Tag zu Reader vorteilhaft, da auch hier eine hohe Erkennungsrate gewährleistet werden kann.

B. Erfassen eines nicht bewegten Gen 2 Passiv Tags, befestigt am Plastikschild eines Carbonhelms

Dieser Test zielt primär auf eine praxisnahe Erfassung der Tags. Hierfür wurde der RFID Reader für diese Messung einmal auf einer Höhe von 130 cm und einmal auf 80 cm befestigt. Der Carbonhelm wurde von einer 180 cm großen Person getragen. Der Helm wurde bei den Messungen nicht bewegt, die Messpunkte befinden sich in einem Abstand von Ein-Meter-Schritten, vom Reader bis zu vier Metern entfernt und jeweils um 10 Grad versetzt von -40° bis $+40^\circ$ Grad. Da der Helm nicht symmetrisch ist, wurde für diesen Testfall der vollständige Öffnungswinkel gewählt. Analog zum vorhergehenden Test wurden pro Messpunkt stets drei Messungen durchgeführt. Dies ergibt für Test 2 eine Gesamtanzahl von 216 Messungen (9 Grad Schritte \cdot 4 Distanzen \cdot 2 Reader-Höhen \cdot 3 Messungen). Die Grafik 3 visualisiert die Auswertung aus Test 2. Die Pfeilrichtung gibt die Richtung des Helms an, wobei die Messergebnisse zeigen, dass die Erfassung des Tag gut möglich ist, wenn dieser nicht vom Helm selbst verdeckt wird. In diesem Zusammenhang sei darauf verwiesen, dass Helme aus Carbon sich störend auf die Funktion des Lesegeräts auswirken [5]. Dies wird an den Messungen bei 3 bis 4 Metern deutlich. Bei -40° kann der Tag gut gelesen werden, da der Reader freie „Sicht“ hat. Bei 40° ist das Plastikschild sowie der Tag durch den Helm verdeckt.

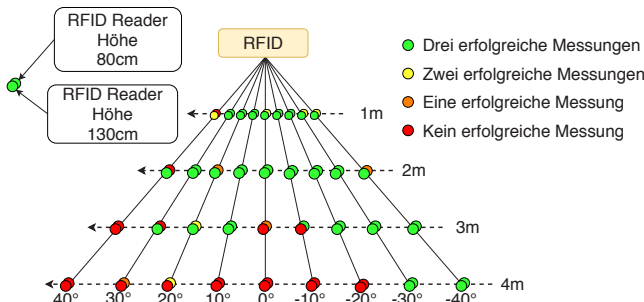


Abbildung 3: RFID Messung mit Carbonhelm

Die Messungen im Bereich von 1 bis 2 Metern sind überwiegend erfolgreich, da der Öffnungswinkel hier den Tag noch nicht beeinträchtigt. Die Ergebnisse aus Test 2 bestätigen ebenfalls den Einfluss der Höhendifferenz. Die Höhendifferenz zwischen Reader und Tag hat einen positiven Effekt, wobei es wenig Unterschied macht, ob die Differenz 50 cm oder 100 cm beträgt.

Das Fazit, das sich aus diesen Messwerten in Bezug auf eine optimale Montage ergibt, stellt sich folgendermaßen dar: Es ist besser, den Fahrer vor dem Scheitelpunkt des Readers zu erfassen, oder, alternativ, den Reader um 1 bis 2 Meter hinter der Lichtschranke zu positionieren. Um Verschattungsprobleme des Carbonhelms [5] zu minimieren muss der Tag am Plastikschild des Helmes angebracht werden.

C. LoRa Geländetest

Mit diesem Test sollte festgestellt werden, ob die Funkübertragung mit LoRa Modulen in alpinem Terrain zuverlässig betrieben werden kann. Gemessen wurde an fünf Positionen, jeweils von einem der Messpunkte zu einem stationär angebrachten LoRa Modul des Typs E45-TTL-1W. Gesendet wurde mit einem E45-TTL-1W sowie mit dem E45-TTL-100 Modul. Von den Messpunkten wurde eine Nachricht an das stationäre Modul gesendet. Wenn dieses die Nachricht erhält, wird sie zurückgeschickt. Somit ist sichergestellt, dass der Verbindungsaufbau in beide Richtungen fehlerfrei funktioniert. Abbildung 4 zeigt die Übertragungstrecken im Gebiet der Ammergauer Alpen [9]. Die Tabelle I gibt eine Übersicht der

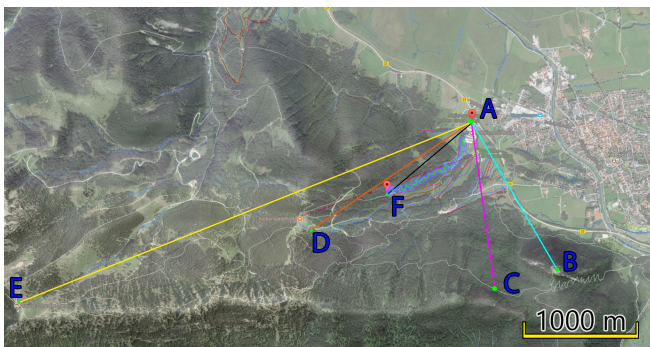


Abbildung 4: Position der Messpunkte auf einem Satellitenbild

Messergebnisse. Eine Messung wird als erfolgreich bewertet, wenn bei jedem Sendeversuch eine fehlerfreie Antwort empfangen wurde. Diese Messergebnisse sind im Zusammenhang

Route	Eigenschaften	Distanz Luftlinie	1 W → 1 W	100 mW → 1 W
B,A	direkte Sichtverbindung	1,49 km	Übertragung erfolgreich	Übertragung erfolgreich
C,A	dichter Wald auf kurzer Strecke	1,4 km	Übertragung erfolgreich	Übertragung erfolgreich
D,A	kein Wald aber keine Sichtverbindung	1,86 km	Übertragung erfolgreich	Übertragung fehlgeschlagen
E,A	kein Wald aber keine Sichtverbindung	4,52 km	Übertragung fehlgeschlagen	Übertragung fehlgeschlagen
F,A	dichter Wald	1,03 km	Übertragung erfolgreich	Übertragung fehlgeschlagen

Tabelle I: Tabelle der Messergebnisse für Funkübertragung im Gelände

mit dem geplanten Einsatzgebiet sehr zufriedenstellend. Die Übertragungsrate war bei diesem Test hoch eingestellt, weshalb durch deren Reduzierung eine höhere Zuverlässigkeit, insbesondere in Waldgebieten, zu erwarten ist.

VI. AUSBLICK

Das Zeiterfassungssystem arbeitet auch über längere Zeiträume zuverlässig. Dennoch sind Verbesserungen in Planung. Diese sind in Soft- sowie auch in Hardware vorgesehen. Zur Minimierung von Clockdrifteffekten über längere Zeiträume wird eine GPS basierte Synchronisation entwickelt. Hinsichtlich der Software ist eine Wartungsfunktion über die LoRa Funkschnittstelle im Aufbau. Dadurch soll ermöglicht werden, den Status der einzelnen Zeitmessstürme über das Internet abzurufen. Eine Kombination mit einem Fahrdynamiksystem ist ebenfalls vorgesehen, damit soll es möglich sein das Fahrwerk des Fahrrades für die jeweilige Strecke und Streckenzustand zu optimieren.

LITERATUR

- [1] Timing Systeme von TagHeuer, www.tagheuer-timing.com/de.
- [2] R. Lacoste, *LoRa Niedrige Geschwindigkeit, hohe Reichweite*, Elektor Nr.555, 2017.
- [3] Bundesnetzagentur, *Vfg 5/2018 SDR*, 2018.
- [4] Semtech Corporation, www.semtech.com.
- [5] Fraunhofer Research News, *Using RFID for fiber composites*, Juli 2013.
- [6] Desing u. Elektronik, <http://www.elektroniknet.de/design-elektronik/industrial-internet-industrie-4/loras-ohne-wan-139548-Seite-3.html>.
- [7] Kurzstreckenfunk, https://de.wikipedia.org/wiki/Short_Range_Devices.
- [8] Realterm: Serial Terminal, <https://sourceforge.net/projects/realterm/>.
- [9] Karte der Ammergauer Alpen, <https://www.google.de/maps/@47.5902804,11.0453458,13z>.
- [10] Chafon RFID Technik, www.chafon.com.
- [11] V. Chawla und D. S. Ha, *An overview of passive RFID*, IEEE Communications Magazine, 2007, 45, 9, 11–17, September 2007.
- [12] M. R. Seye und B. Gueye und M. Diallo, *An evaluation of LoRa coverage in Dakar Peninsula*, Proc. of the 8th IEEE Electronics and Mobile Communication, pp. 478–482, Oktober 2017.
- [13] A. M. Baharudin und W. Yan, *Long-range wireless sensor networks for geo-location tracking: Design and evaluation*, Proc. Int. Electronics Symp. (IES), pp. 76–80, September 2016.

Demo: An Ontology-based NETCONF-MQTT Bridge for Sensor Devices in the IoT

Kristina Sahlmann¹, Alexander Lindemann², Thomas Scheffler³, Bettina Schnor⁴

¹University of Potsdam, Potsdam; HTW Berlin, Berlin, Germany, Email: sahlmann@uni-potsdam.de

^{2,4}University of Potsdam, Potsdam, Germany, Email: allindem | schnor@cs.uni-potsdam.de

³Beuth-Hochschule für Technik, Berlin, Germany, Email: scheffler@beuth-hochschule.de

Abstract—This demo shows how IoT devices can be managed by the network configuration protocol NETCONF. We decouple the NETCONF server from the device through a MQTT broker. For the semantic interoperability, we use ontology-based device descriptions. This way, devices can dynamically join a network and announce their capabilities. Furthermore, we implemented a web interface for the NETCONF client to display device capabilities and which can be used for device control.

I. INTRODUCTION

This demo presents an extension to an experiment by Scheffler and Bonneß [1]. It was their aim to manage dynamic IoT networks using a standards based approach. They developed a NETCONF-MQTT bridge which translates between the IoT specific MQTT [2] and the NETCONF [3] protocol using a dynamically generated YANG [4] model.

In contrast to the original experiment which used a proprietary format for the description of device capabilities, we use the oneM2M Base ontology [5] for the same purpose. While in the original experiment a LIFX LED bulb was controlled via WLAN, we integrate a microcontroller board with LEDs which is connected by 6LoWPAN [6] to the network (for details and evaluation see [7], [8]).

The Network Configuration Protocol (NETCONF) is a new standard for IP-based networks defined by the IETF which provides mechanisms to install, manipulate, and delete the configuration of network devices. The NETCONF protocol uses the YANG data modeling language to specify data models and protocol operations. The NETCONF protocol operations are realized as remote procedure calls (RPCs).

The MQTT protocol was developed by IBM and later standardized by Oasis. MQTT is a Machine-to-Machine (M2M) protocol and applies the Publish/Subscribe pattern for data dissemination in distributed systems. Messages reference so-called *Topics* of interests.

II. DEPLOYMENT SETUP

As shown in Figure 1, our system architecture consists of four components. The end node is a CC2538EM microcontroller board from Texas Instruments (TI)¹. The TI-board uses an ARM Cortex M3 processor and supports the IEEE 802.15.4 standard. The TI-Board is a very constrained device with 32 kB RAM and 512 kB Flash memory. The TI-Board

uses Contiki OS² and its implementation of MQTT³.

Further, there are the typical MQTT instances: a broker and two clients (the microcontroller board and the NETCONF server). The MQTT broker manages publication and subscription between the TI-Board and the NETCONF server. The MQTT broker runs Mosquitto⁴. The NETCONF server is responsible for the network management. A user can control the TI-board via a NETCONF client.

The NETCONF-MQTT bridge translates the specific device descriptions into a dynamic YANG model. Further, it translates the RPC calls into MQTT messages. The software architecture of the NETCONF-MQTT bridge is shown in Figure 2. The implementation of the bridge is based on existing Python libraries: paho-mqtt⁵ for the MQTT client, pyang⁶ for the YANG generator, netconf⁷ for the NETCONF server.

The NETCONF client is implemented in Python and uses ncclient⁸ library and micro-framework Flask⁹ for the web interface.

III. SEMANTIC INTEROPERABILITY

We have chosen the oneM2M Base Ontology [5] (from now on referred to as oneM2M ontology) for two reasons: (i) it is a small ontology for service and functionality description of devices; and (ii) it is represented by the OWL standard. The oneM2M ontology offers a vocabulary for the description of application services and device functionality. When all devices in a network use this ontology for describing themselves, a query can easily extract necessary information.

First, a device profile (based on the ontology) must be created (e.g. using a tool like Protégé¹⁰) and deployed on the TI-board. Currently, we must also configure the MQTT broker address, because Contiki has limited support for multicast. We implemented three use cases which are shown in Figure 1:

- **Device Capabilities Discovery:** (1) the TI-board publishes its device description to the MQTT broker's

²<http://contiki-os.org/>

³<https://github.com/contiki-os/contiki/tree/master/apps/mqtt>

⁴<https://mosquitto.org>

⁵<http://www.eclipse.org/paho/clients/python/docs/>

⁶<https://github.com/mbj4668/pyang>

⁷<https://github.com/choppsv1/netconf>

⁸<https://github.com/ncclient/ncclient>

⁹<http://flask.pocoo.org/>

¹⁰<http://protege.stanford.edu/>

¹<http://www.ti.com/tool/cc2538dk>

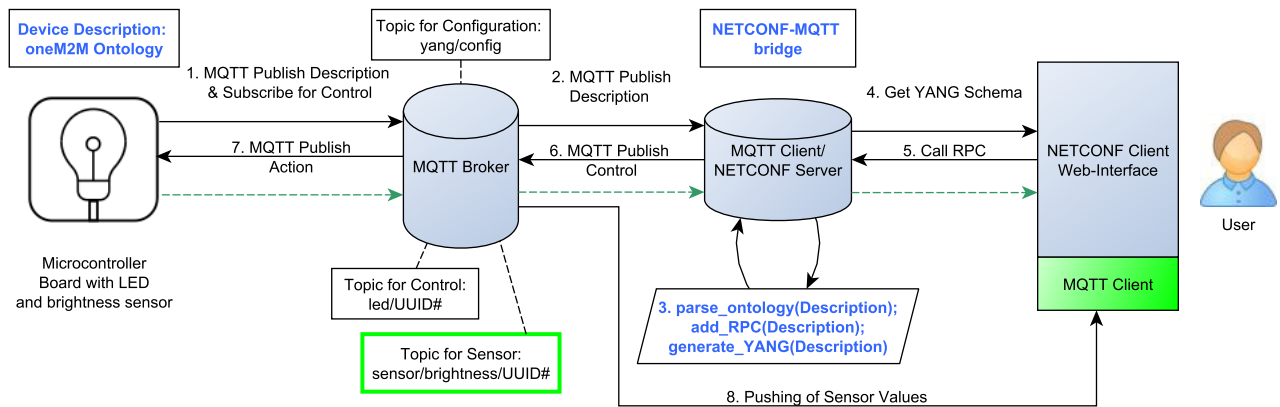


Fig. 1. System-Architecture with MQTT and NETCONF [8]

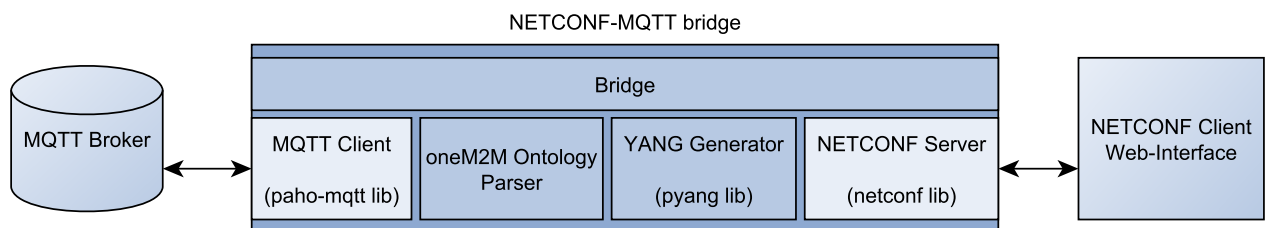


Fig. 2. NETCONF-MQTT bridge software architecture [8]

- Topic "yang/config"; (2) the NETCONF server will be notified about this publication; (3) the device description will be parsed using ontology information, an RPC added to the NETCONF server and the YANG model generated; (4) a user retrieves device capabilities via web interface.
- **Actuator Scenario:** (5) a user executes an RPC call for a control functionality (e.g. switch on an LED in green); (6) this RPC call will be published as a MQTT message; (7) and the TI-board will be notified; the action executed and a response will be send back to the user (green dashed arrows).
 - **Sensor Scenario:** (8) the brightness sensor publishes values periodically to the MQTT broker; the MQTT client in the web interface is subscribed to this Topic directly because YANG datastore subscription is still a draft [9].

IV. CONCLUSION

This demo shows how sensor devices in the IoT can be managed with the standard network management protocol NETCONF. Device capabilities are described with the help of the oneM2M ontology and published via a MQTT broker to the network.

ACKNOWLEDGMENT

This research is funded by the Federal Ministry of Education and Research under grant number Professors Program II. The

responsibility for the content of this publication lies with the authors.

REFERENCES

- [1] T. Scheffler and O. Bonneß, "Manage resource-constrained IoT devices through dynamically generated and deployed YANG models," in *Applied Networking Research Workshop (ANRW) 2017*. IETF, 2017.
- [2] "MQTT Version 3.1.1," 2014. [Online]. Available: <http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html>
- [3] "RFC 6241 - Network Configuration Protocol (NETCONF)," 2011. [Online]. Available: <https://tools.ietf.org/pdf/rfc6241.pdf>
- [4] "RFC 6020 - YANG - A data modeling language for NETCONF," 2010. [Online]. Available: <https://tools.ietf.org/pdf/rfc6020.pdf>
- [5] "Base Ontology: oneM2M Technical Specification: TS-0012-V2.0.0," 30 August 2016. [Online]. Available: <http://www.onem2m.org/technical/published-documents>
- [6] "RFC 4944 - Transmission of IPv6 Packets over IEEE 802.15.4 Networks," 2007. [Online]. Available: <http://tools.ietf.org/html/rfc4944>
- [7] K. Sahlmann, T. Scheffler, and B. Schnor, "Managing IoT device capabilities based on oneM2M ontology descriptions," in *Proceedings of the 16. GI/ITG KuVS Fachgespräch Sensornetze*, ser. Technical Reports. Hamburg, Germany: HAW Hamburg, 2017, pp. 23–26.
- [8] —, "Ontology-driven device descriptions for iot network management," in *2018 IEEE Global Internet of Things Summit (GIoTS) Proceedings, 3rd Workshop on Interoperability and Open-Source Solutions for the Internet of Things (InterOSS-IoT 2018)*. Piscataway, NJ, USA: IEEE, 2018, pp. 353–358.
- [9] A. Clemm, E. Voit, A. Prieto, A. Tripathy, E. Nilsen-Nygaard, A. Bierman, and B. Lengyel, "YANG Datastore Subscription:" Working Draft, IETF Secretariat, Internet-Draft draft-ietf-netconf-yang-push-15, February 2018.

Demo: A Case for Chirp Modulation for Low-Power Acoustic Communication in Shallow Waters

Fabian Steinmetz, Jan Heitmann, Christian Renner

Research Group smartPORT

Hamburg University of Technology

Email: {fabian.steinmetz, jheitmann, christian.renner}@tuhh.de

Abstract—Small and cheap micro AUVs enable diverse underwater monitoring applications in shallow inshore waters; e.g., inspection of underwater assets, observation of water quality, and identification of pollution sources. The formation and collaboration of swarm members yet requires communication and self-localization based on cheap, miniature acoustic devices. However, this is severely hampered by the effects of multi-path propagation in shallow waters. We study the benefits and applicability of narrow-band chirp-based modulation vs. frequency-shift keying (FSK).

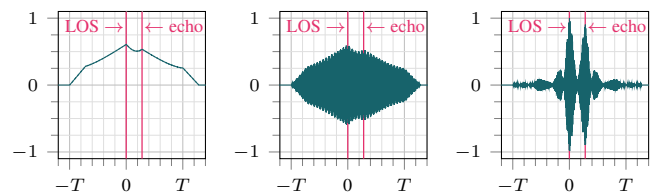
I. INTRODUCTION

Exploration and monitoring of underwater sceneries is drawing considerable attention. Recent examples are the investigation of sub-mesoscale eddies [1] or ship tracking in harbors [2]. Timely acquisition of data mandates communication—typically wireless to keep installation and maintenance cost low. Shallow and relatively small water bodies—such as port basins, lakes, or canals—are a particularly challenging scenery. Reflections at the water surface cause massive inter- and intra-symbol interference, because the non-line-of-sight (NLOS) signals have a low delay and attenuation only [3]. In previous research [4] and recent measurement campaigns, we experienced significant attenuation and amplification of frequency shares depending on position and time. In addition, we noted influence of environmental conditions and their change over time. As a counter measure we embark on the use of narrow-band chirp signals for preamble-based synchronization, motivated by the following observations. We noted that communication with our acoustic modem was—despite using FSK modulation—reliable, if the preamble was successfully detected by the receiver. Failure to do so arises mainly from cancellation and amplification caused by reflections and scattering. The main reason for this is intra-symbol interference leading to unreliable symbol detection and hence synchronization. Once the synchronization has succeeded, however, the symbol windows are known and reliable communication can be achieved through relatively simple methods such as frequency hopping and redundancy coding.

II. FUNDAMENTALS

A. Frequency-Shift Keying

Frequency-Shift Keying (FSK) is commonly used in acoustic underwater communication. In binary frequency-shift keying (BFSK), a bit b is transmitted as a sinusoidal symbol with



(a) FSK non-coherent (b) FSK cross-corr. (c) chirp cross-corr.

Fig. 1: Detection of chirp ($f_s = 50$ kHz, $f_e = 54$ kHz) and FSK ($f = 50$ kHz) signal superimposed by an echo with 0.7 ms delay and 90% amplitude (ca. 10 m distance at 2 m depth). Symbol duration is $T = 2.5$ ms. Time on the x-axis is relative to perfect LOS detection.

frequency f_b and duration T . There are more complex forms of FSK, which we do not address in detail due to space constraints. Receivers often employ non-coherent detection due to its efficiency, but cross-correlation is also possible. Orthogonal frequencies f_b may improve detection. The (envelope) shape of detector output is trapezoidal for undistorted symbols with constant amplitude, with its peak marking the symbol's end. If reflections overlap with the line-of-sight (LOS) signal, detector output contains overlapping triangles as in Figs. 1a and 1b. Depending on the number and phase of reflections, a clear peak is no longer present, impacting detection accuracy and likelihood.

B. Chirp Keying

A chirp is a signal with steadily changing frequency over time. We consider linear chirps, where the frequency sweeps linearly over time from frequency f_s to f_e . The chirp has duration T and bandwidth $B = |f_e - f_s|$. A bandwidth-efficient way to employ binary modulation is to use down-chirps ($f_s > f_e$) and up-chirps ($f_s < f_e$) to represent bit b . Detection of a chirp can be achieved through cross-correlation. The detector output is a narrow and steep peak, which allows a clear distinction between the LOS signal and reflected signals as displayed in Fig. 1c. Because of this property, chirps are commonly employed in radar applications.

C. Comparison and Discussion

In various real-world experiments [4], we confirmed heavy, time-varying, and location-dependent frequency selectivity of the acoustic channel. This is of paramount relevance for

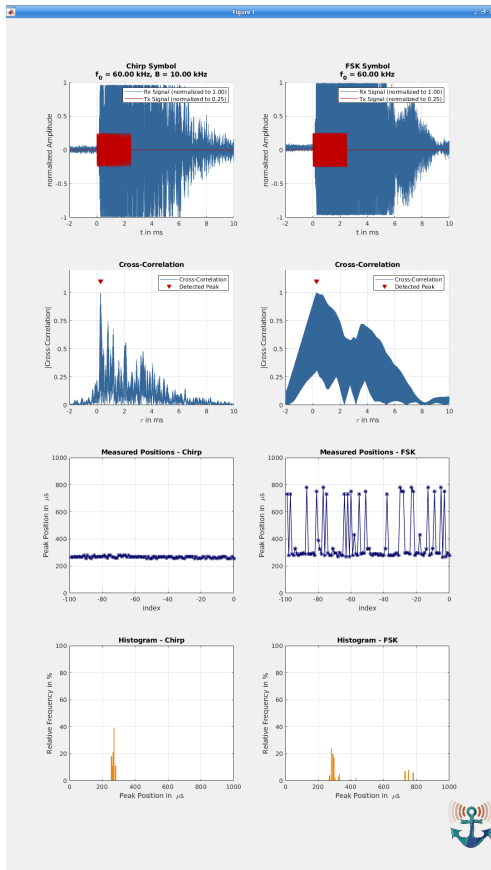


Fig. 2: User Interface of our MATLAB implementation for demo purposes.

preamble-based synchronization. Here, symbol (window) positions are derived by detecting the preamble symbols or their peaks, respectively. For successful packet reception and precise time-of-flight ranging [5], accurate synchronization is mandatory. In case of FSK, signal cancellation and reverberation due to reflections cause poor synchronization. As shown in Fig. 1, the peak of the originally trapezoidal detection result is blurred and accurate detection hampered. Noise and multiple echos aggravate this situation. In case of destructive interference, a peak may not be found at all.

Chirp modulation brings two advantages: (i) its frequency spreading increases detection probability, because attenuation of frequency shares due to reflection is drastically reduced, and (ii) the peaks of the LOS signal and its echos are separable in the time domain, so that a smaller synchronization error is expected. Using chirps yet comes at a cost. The benefit of non-coherent FSK detection is its low computation complexity. For each new sample, two multiplications are required, giving a constant computation complexity per new sample. Without optimization, cross-correlation has to be performed for a full symbol for every sample, yielding at least logarithmic computation complexity per new sample. For low-power acoustic modems, this complexity may already be infeasible. Hence, we study the performance of chirp vs. FSK in general but also discuss the feasibility of chirp detection on low-power

acoustic modems.

III. DEMO SETUP

Details on our implementation and evaluation of chirp keying compared to frequency shift keying are given in the accepted paper. For the demo itself, we use the smartPORT acoustic modem [4], a low-power, low-cost device for use in μ AUVs such as Hippocampus [6]. We deploy two hydrophones in a small glass tank with a distance of several centimeters. We send a single symbol, an up-chirp or a FSK symbol. The base frequency and the bandwidth, in the case of an up-chirp, can be adjusted by the visitor during the demo.

Signal generation is done with MATLAB and an external USB oscilloscope and waveform generator TiePie HS5. The generator is connected to the input of the transmit circuitry of the modem. It is amplified and fed to the hydrophone.

The signal, received from the second hydrophone is amplified by the receiver circuit of our modem. We remove the band pass filters to eliminate any influence on the received signal. The output of the analog processing chain is captured with the oscilloscope and plotted in comparison to the transmitted signal with MATLAB.

In addition to that, we show the result of cross-correlation with a stored reference signal to explain the better separability of chirp signals in comparison to a FSK signal. The peak position in the time domain, detected by our synchronization algorithm is also shown for both modulations and indicates a lower variation using chirp modulation. The interface we use for the demonstration is depicted in Fig. 2.

ACKNOWLEDGMENT

This work has been partially supported by the German Federal Ministry of Education and Research (BMBF, FKZ 13N14153), the German Federal Ministry for Economic Affairs and Energy (BMWi, FKZ 03SX463C), and ERA-NET Cofund MarTERA (contract 728053).

REFERENCES

- [1] B. Meyer, C. Isokeit, E. Maehle, and B. Baschek, "Using Small Swarm-Capable AUVs for Submesoscale Eddy Measurements in the Baltic Sea," in *Proc. of the MTS/IEEE Oceans Conf.*, Sep. 2017.
- [2] M. Rao, N. K. Kamila, and K. V. Kumar, "Underwater Wireless Sensor Network for Tracking Ships Approaching Harbor," in *Proc. of the IEEE International Conf. on Signal Processing, Communication, Power and Embedded System (SCOPE5)*, Oct. 2016.
- [3] T. Jensenud and S. Ivansson, "Modeling the Power Delay Profile of Underwater Acoustic Channels — The Effects of Out-of-Plane Scattering and Reverberation," in *Proc. of the International Conf. Underwater Communications and Networking (UComms)*, Sestri Levante, Italy, Sep. 2014.
- [4] C. Renner and A. J. Golkowski, "Acoustic Modem for Micro AUVs: Design and Practical Evaluation," in *Proc. of the 11th ACM International Conf. on Underwater Networks & Systems (WUWNet)*, Shanghai, China, Oct. 2016.
- [5] C. Renner, "Packet-Based Ranging with a Low-Power, Low-Cost Acoustic Modem for Micro AUVs," in *Proc. of the 11th International ITG Conf. on Systems, Communications and Coding (SCC)*, Hamburg, Germany, Feb. 2017.
- [6] A. Hackbarth, E. Kreuzer, and E. Solowjow, "HippoCampus: A Micro Underwater Vehicle for Swarm Applications," in *Proc. of the IEEE/RSJ International Conf. on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, 2015.

Demo: An Interactive Demonstration of the PotatoScanner, a Mobile DTWSN Node

Björn Gernert

*Institute of Operating Systems and Computer Networks
Technische Universität Braunschweig
Braunschweig, Germany
gernert@ibr.cs.tu-bs.de*

Lars Wolf

*Institute of Operating Systems and Computer Networks
Technische Universität Braunschweig
Braunschweig, Germany
wolf@ibr.cs.tu-bs.de*

Abstract— In recent years, research has increasingly focused on monitoring crops by means of sensors on agricultural land. Smart Farming or Precision Farming helps to ensure that plants have an optimal environment for their development. Besides the collection and aggregation of the sensor data, the transmission is an critical point.

The aim of the PotatoScanner [1] is to act as a Delay Tolerant Wireless Sensor Network (DTWSN) node and to measure the leaf temperature of plants in the field. The setup demonstrates the functionality of the PotatoScanner with a model. Surface temperatures of any objects can be measured, which are displayed on a screen and visualized with the help of RGB-LEDs. The sensors and the visualization are performed by the same hardware that is used in the real PotatoScanner. In addition, the determined data is transmitted to a tablet via Delay and Disruption Tolerant Network (DTN) and can also be visualized on it.

Index Terms—DTWSN; delay tolerant networking, field sprayer; potatoes;

I. INTRODUCTION

Figure 1 shows the setup of the demo. It consists of a smaller version of the sensor system of the PotatoScanner (left) and a sensor node, which normally is mounted on the field sprayer (right).

The assembly on the left is additionally equipped with several RGB LEDs that assign a color value to the currently measured temperature. If a visitor holds his hand under the corresponding sensor, the color changes. At the same time, the measured temperature can be displayed on a screen (not in the figure).

The box of the sensor node is open. The node consists of a Raspberry Pi, a Powerbank (battery) and a GPS module. The sensor node is supplied with power via the Powerbank.

On the already mentioned screen you can see the temperature and some pictures of the setup as well as pictures of the evaluation.

II. FUNCTION OF THE REAL POTATOSCANNER

The PotatoScanner consists of several subsystems, which together allow the monitoring of agricultural areas. Small wireless sensor nodes are deployed on the field first. This can already be done during sowing. These small sensor nodes (shown as yellow circle in Figure 2) collect measurement data such as soil temperature and soil moisture.



Fig. 1: Demo setup of the PotatoScanner

In a first step, this data is transmitted to larger nodes at periodic intervals (a). These larger nodes are equipped with a solar module and a large battery, so that they can always be accessed by the smaller nodes. In addition, the larger node consists of two microcontrollers, a tiny one and a big one. The tiny microcontroller is operated continuously and can switch the bigger one on and off as required. Furthermore, it can communicate with the small sensor nodes deployed on the field and receive their measurement data. The bigger microcontroller has significantly more computing power and can communicate with the field sprayer [2] via WiFi.

When the farmer drives with the field sprayer over the field,

Demo: InPhase – 3D Localization in Wireless Sensor Networks

Yannic Schröder

Institute of Operating Systems and Computer Networks
Technische Universität Braunschweig
Braunschweig, Germany
schroeder@ibr.cs.tu-bs.de

Lars Wolf

Institute of Operating Systems and Computer Networks
Technische Universität Braunschweig
Braunschweig, Germany
wolf@ibr.cs.tu-bs.de

Abstract—The location of nodes in Wireless Sensor Networks can be valuable information, e.g. for routing sensor data to a sink node. We demonstrate the InPhase localization system. It allows to locate wireless sensor nodes in 3D space. By measuring the phase response of the radio communication channel the distance between two sensor nodes can be computed. With multiple distance measurements the location of a wireless sensor node can be determined. The measurement is done via a standard IEEE 802.15.4 radio transceiver. No extra hardware besides the radio is required.

Index Terms—localization, ranging, distance estimation, WSN, phase-based

I. INTRODUCTION

Wireless sensor nodes can measure a large variety of different data, e.g. temperature, humidity or acceleration. However, obtaining the location of sensor nodes is problematic. While inexpensive sensors with low power consumption can be found for the above mentioned environmental data, location sensors are only available as Global Navigation Satellite System (GNSS) receivers. These devices have some caveats: They are rather expensive, need an extra antenna and have a high power consumption. Further, GNSS receivers only work outdoors, as signals from the satellites need to be received.

We demonstrate the InPhase localization system. It allows to localize nodes in a Wireless Sensor Network (WSN) by only employing an already available IEEE 802.15.4 radio transceiver. The system measures the phase response of the 2.4 GHz radio channel between two sensor nodes and computes the physical distance. From multiple distance measurements to sensor nodes in known locations, the position of a sensor node with unknown location is determined.

II. DISTANCE ESTIMATION

The distance estimation is implemented according to our previously published work [1], [2]. A distance is always measured between two nodes of the WSN. The phase response of the radio channel between the two nodes is measured via the Active-Reflector-Principle (AR-Principle) [3]. The AR-Principle requires radio transceivers that can measure the phase of an incoming radio signal. We employ off-the-shelf AT86RF233 radio transceivers with integrated Phase Measurement Units (PMUs) [4]. Figure 1 shows an example phase response of such a measurement. The result resembles a

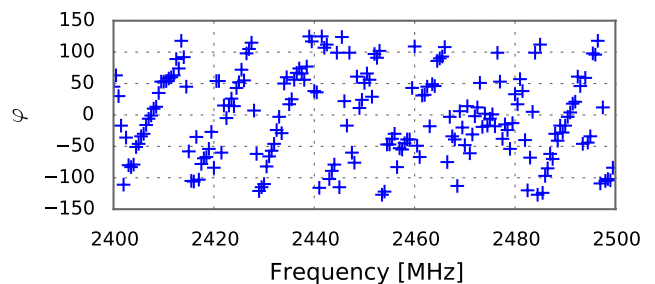


Fig. 1. Raw phase response measurement (reproduced from [1])

sawtooth signal. The steepness of the slope is proportional to the distance between sensor nodes. The Complex-valued Distance Estimation (CDE) algorithm is used to determine the distance from this raw measurement data [2].

III. LOCALIZATION

For localization, multiple distances are measured (see Section II). Multiple sensor nodes with known locations – so called anchors – are deployed in an area. A sensor node with unknown location measures distances to the anchors in a round-robin fashion. The measured data is relayed to a server that computes the distance and localizes the sensor node.

The localization problem is solved by a particle filter in multiple localization rounds. Each particle represents a potential location of the sensor node. Initially, when the location is unknown, all particles are randomly distributed in the whole 3D volume of the allowed area.

Then, for each measured distance to an anchor node, the probability of every particle is determined. If the particle's distance to the anchor fits the measurement, it is marked with a high probability. If its distance to the anchor node does not match the measured distance, it is marked with a low probability. Afterwards, a final location for this localization round is derived from the weighted particle cloud.

The particle positions are then refined based on the assigned probabilities. Particles with low probability will be removed, particles with high probability will be cloned and considered as possible locations for the round. Before the localization loop is restarted with the next distance measurement, all particles are moved randomly in a small radius to allow the cloned particles to spread out.

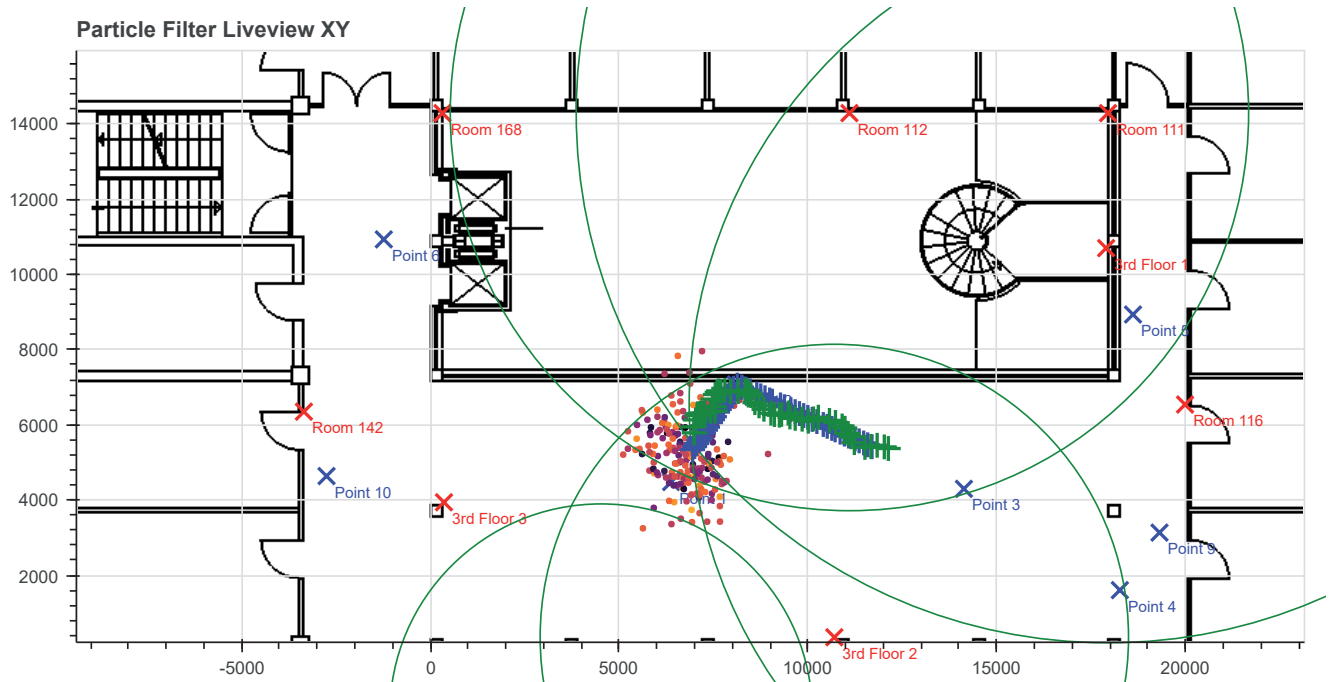


Fig. 2. Visualization of the localization algorithm. Red x's: Anchor nodes. Blue x's: Known reference points. Green Circles: Measured distances. Blue +symbols: Ground truth path. Green +symbols: Localization output. Orange point cloud: Particles. Both axes are marked in millimeters. Reproduced from [5].



Fig. 3. Battery-powered InPhase sensor node (reproduced from [5]).

IV. DEMO SETUP

The live demonstration shows a full localization setup that localizes a wireless sensor node with unknown location (tag) inside the demo area. Our InPhase sensor nodes are used as anchors and tag, see Figure 3.

The anchor nodes are mounted to walls or beams in the deployment area. To allow 3D localization, the anchors are spread over multiple different heights across multiple floors. Generally, the anchor nodes need a Line-of-Sight (LOS) to the tag node for the measurement to succeed. However, as this cannot always be guaranteed in real deployments, the demo area features some obstacles that introduce Non-Line-of-Sight (NLOS) conditions. This allows experimentation with adverse channel conditions during the live demonstration.

The tag node is mounted on a tripod and can be freely moved in the demo area. An attached single-board computer relays the measurement data to a server. The server runs the localization algorithm and displays the resulting location in real-time, see Figure 2.

REFERENCES

- [1] G. von Zengen, Y. Schröder, S. Rottmann, F. Büsching, and L. C. Wolf, "No-Cost distance estimation using standard WSN radios," in *The 35th Annual IEEE International Conference on Computer Communications (INFOCOM 2016)*, San Francisco, USA, Apr. 2016.
- [2] Y. Schröder, D. Reimers, and L. Wolf, "Accurate and precise distance estimation from phase-based ranging data," in *2018 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, Sep. 2018.
- [3] W. Kluge and E. Sachse, "System, method, and circuit for distance measurement between two nodes of a radio network," Feb. 2014, US Patent 8,644,768.
- [4] *Low Power, 2.4GHz Transceiver for ZigBee, RF4CE, IEEE 802.15.4, 6LoWPAN, and ISM Applications*, Atmel Corporation, San Jose, Jul. 2014.
- [5] Y. Schröder, D. Reimers, and L. Wolf, "Demo: InPhase – no-cost phase-based ranging and localization in wireless sensor networks," in *3rd KuVS/GI Expert Talk on Localization*, Jul. 2018.

Demo: BATS a Flexible System for Automated Encounter Detections

Björn Cassens
TU Braunschweig
Brunswick, Germany
cassens@ibr.cs.tu-bs.de

Rüdiger Kapitza
TU Braunschweig
Brunswick, Germany
kapitza@ibr.cs.tu-bs.de

Abstract—Tracking animals is mandatory for a better understanding of their social interactions. Social behavior of bats is of particular interest, since it can be used for improved epidemic avoidance and forecasts. Animal-borne sensor nodes are enabling the monitoring and analysis of social contacts. Attached to bats, these nodes can only be equipped with small batteries, dictating tight requirements regarding the energy demand of the soft- and hardware infrastructure.

In our demo, we present our software and hardware architecture of a wireless sensor node that allows a fully-automated tracking of the social behavior of free ranging animals. This system underwent continuous improvements leading to several prototypes which are presented in our demo. In order to showcase that our system works under realistic conditions, we ran several field tests in the tropical rain forest of Panama and multiple tests in Germany. We present some of our results from those field tests and give in-depth insights of our software.

Index Terms—Demo, Animal Tracking, Energy Aware Systems, Sensor Networks

I. INTRODUCTION AND MOTIVATION

The observation of the behavior of animals in their natural environment is a great challenge for biologists. In the daily work of the biologists, they rely on technical solutions to study wildlife in detail since direct observation may not be feasible for cryptic or highly mobile species like bats. Wireless sensor network consisting of miniaturized animal-borne sensor nodes (called mobile node) give several new opportunities for wildlife tracking as prior works shows [13], [14]. Wireless sensor nodes are not only able to collect and store physiological or environmental data, they also allow an automated remote download of data. Therefore, extensive data sets can be created with moderate effort due to a high degree of automation while also minimizing impacts on the observed animal. Thus an almost unbiased behavior can be expected.

Due to an increased functionality of the nodes, a higher energy demand compared to the state of the art radio telemetry is the result. Current high-performance tracking devices carry large batteries to reach a decent runtime and are in turn rather heavy (the weight ranges from 30 g [13] up to 200 g [14]). As an animal-borne sensor node is not allowed to exceed 5-10% of the observed animal's body weight [12], those systems can only be used for large-sized vertebrates. In our project, we use sensor nodes, with a weight of less than 2 g including housing and battery. Therefore, only tiny energy budgets are possible to ensure a runtime of 14 days which

requires sophisticated software architecture for energy efficient on-board data collection.

Our goal to track animals and especially bats is because bats form the second largest group of mammals with more than 1000 species while the majority is living in groups. Group size may vary depending on the species from few individuals to several millions and social systems may vary strongly in their complexity [11]. Furthermore, bats play central roles in ecosystems by providing crucial services such as pollination, seed dispersal, and pest control [10], but at the same time many species are endangered [9]. Even though detailed knowledge on foraging strategies is scarce, there is increasing evidence that social interactions contribute to foraging success [7], [8]. More recently, bats also received negative publicity for being reservoirs of many emerging infectious diseases and being involved in spillovers to humans and livestock [5], [6]. Detailed knowledge about (social) behavior is crucial to understand the interdependencies and dynamics within bat populations and with their environment. This knowledge in turn is key to develop successful conservation strategies in order to preserve the valuable services provided to humans and to understand dynamics of infectious diseases to prevent future outbreaks [4].

A common way of observing gregarious animals is the use of global positioning system nodes and miniaturized loggers, which are available at a weight of 1 g. However, loggers need to be retrieved for data access and an option for remote download multiplies the weight [3]. A miniaturized 1.3 g version of the 'EncounterNet', an automated solution for encounter logging [2], would be light enough for the application in many bat species, but the tag miniaturization limits the runtime to less than 24h [15]. It is essential to observe a significant share of the individuals of a population over longer time periods at minimal levels of disturbance in order to draw population (or even species) wide conclusions from behavioral data.

Therefore, systems for automatic data collection of bats must fulfill the particularly challenging task of tracking interactions among group living bats.

- Smart energy management that allows data collection over a period of 1-2 weeks at high sampling rates and at a mobile node weight of < 2 g
- Remote data download which limits disturbances of the animals by the researchers to the tagging event thus allowing for observations on unbiased, natural behavior.
- Full automation of the system and operation of at least 30 mobile nodes at a time, enabling the observation of entire social groups at a fraction of the costs of conventional methods.
- Two-way communication of mobile nodes allowing for direct encounter logging creating dyadic data sets that form the basis of state-of-the-art social network analysis. On board collection and processing of accessory data (maximum received signal strength indicator and duration of encounters) that may be used to weight social networks.

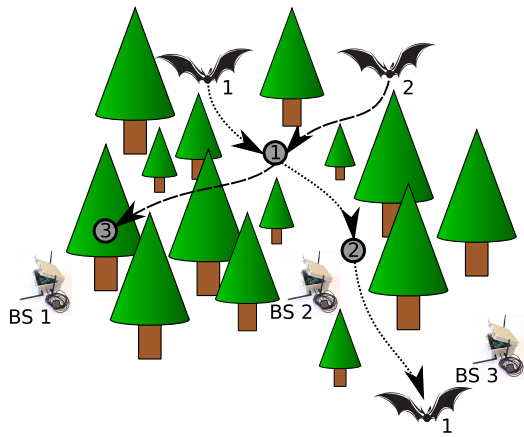


Fig. 1. Overview of the BATS system, consisting of a mobile node and a base station.

II. SYSTEM OVERVIEW

As shown in Figure 1 a deployed system consists of mobile nodes and base stations. Both are interacting closely with each other and its operations are explained in the following. Two bats, outfitted with our mobile nodes and the identifier 1 and 2 are released into the wild. Due to a periodic exchange of so-called mobile node beacons we can detect the presence of one or more bats in communication range. In order to save energy, a wakeup-receiver is used which comes along with a limited communication range of 10 m. Once two bats are in communication range, a meeting is assumed ① and the values like start of an encounter and duration are updated accordingly. If the bats are separating from each other, the meetings are stored internally and are awaiting data transmission.

If a bat is flying by a base station ②, the data is sent to the base station immediately. In the case of a resting bat

inside a roost ③, the frequency of sent mobile-node beacons is decreased to save energy. Also data is not sent directly, as we assume a large number of potential mobile nodes inside a roost. Therefore, data is sent in a time division multiplex access alike scheme to prevent collisions. However, in all cases, data is stored internally of the base stations which can be downloaded on daytime which reduces impacts on the observed animals. Therefore, an unbiased behavior can be expected.

III. RESULTS

Designing such kind of system is a challenge, resulting in multiple prototypes. In our demo, we show multiple stages of our hardware, ranging from first mock-up boards up to deployed nodes. As the software closely interacts with the hardware, we also introduce the software and its properties. We also discuss one of our protocol optimizations, to save up to 40 % per transmitted data point without sacrificing reliability. The results from one field test in the rain forest in Panama 2016 [1] are presented, showcasing that our system is capable to work in real world conditions.

REFERENCES

- [1] B. Cassens, S. Ripperger, M. Hierold, F. Mayer and R. Kapitza; Automated Encounter Detection for Animal-Borne Sensor Nodes; EWSN 2017
- [2] C. Rutz, Z. T. Burns, R. James, S. M. H. Ismar, J. Burt and B. Otis and J. Bowen and J. J. H. St Clair; Automated mapping of social networks in wild birds; Elsevier 2012
- [3] R. Kays, M. C. Crofoot, W. Jetz and M. Wikelski; Terrestrial animal tracking as an eye on life and planet; American Association for the Advancement of Science 2015
- [4] H. Whitehead; Analyzing animal societies: quantitative methods for vertebrate social analysis; University of Chicago Press 2008
- [5] J. F. Drexler, V. M. Corman, M. A. Müller, G. D. Maganga, P. Vallo, T. Binger, F. Gloza-Rausch and others; Bats host major mammalian paramyxoviruses; Nature Publishing Group 2012
- [6] C. H. Calisher and J. E. Childs, H. E. Field, K. V. Holmes and T. Schountz; Bats: important reservoir hosts of emerging viruses; Clinical microbiology reviews 2006
- [7] N. Cvikel K. E. Berg, E. Levin, E. Hurme, I. Borissov, A. Boonman, E. Amichai and Y. Yovel; Bats aggregate to improve prey search but might be impaired when their density becomes too high; Current Biology 2015
- [8] M. T. O'Mara and D. Dechmann and R. Page; Frugivorous bats evaluate the quality of social information when choosing novel foods; Behavioral Ecology 2014
- [9] K. E. Jones, A. Purvis and J. L. Gittleman; Biological correlates of extinction risk in bats; The American Naturalist 2003
- [10] S. Ghanem and C. Voigt; Increasing awareness of ecosystem services provided by bats; Elsevier Science & Technology 2012
- [11] G. Kerth; Causes and consequences of sociality in bats; Oxford University Press 2008
- [12] S.K. Amelon, D.C. Dalton and J.J. Millsbaugh and S.A. Wolf; Radiotelemetry: Techniques and Analysis; The Johns Hopkins University Press 2009
- [13] P. Sommer, J. Liu, K. Zhao, B. Kusy, R. Jurdak, A. McKeown and D. Westcott; Information Bang for the Energy Buck: Towards Energy- and Mobility-Aware Tracking; EWSN 2016
- [14] P. Zhang, C. M. Sadler, S. A. Lyon and M. Martonosi, Margaret; Hardware Design Experiences in ZebraNet; Sensys 2004
- [15] I. Levin, D. M. Zonana, and J. M. Burt and R. J. Safran; Performance of encounter tags: field tests of miniaturized proximity loggers for use on small birds; Public Library of Science 2015

Demo: Combining Temperature-Controlled Chambers to Simulate Energy Harvesting from Thermal Differences

Robert Hartung, Sven Pullwitt and Lars Wolf
Institute for Operating Systems and Computer Networks
Technische Universität Braunschweig
 {hartung | pullwitt | wolf}@ibr.cs.tu-bs.de

Abstract—Our demo presents a testbed to simulate energy harvesting from temperature differences. By combining two temperature-controlled chambers, we can simulate ambient temperature gradients and use the gradient to harvest energy through a thermoelectric element. An electronic load is used simulate a load, which in our case was used to find the maximum power point. We have used the experimental setup to derive models from harvestable energy from a thermo-electric generator. This model has been used for a feasibility study on energy harvesting from soil temperature. We present results shortly and present problems and solutions from our experiments at the end.

I. INTRODUCTION

Energy harvesting is expected to change today's application in both Wireless Sensor Networks (WSNs) and the Internet of Things (IoT) drastically. While traditional networks have often been limited in the amount of energy available, modern networks might be powered fully or partially by harvested energy for months or years. While this extends the lifetime in general, completely powering a network from harvested energy yields new challenges. The available energy is usually very unstable and varying. For solar-powered systems, energy is only to be expected during day-time. Other sources such as harvesting energy from temperature gradients, might have less energy overall, but power can also be expected during night-time. Intermittent Computing addresses these issues by leveraging the execution time from continuous to partially powered sensor nodes. The intermittent execution of sensor nodes leads to several new challenges, such as loss of state or scheduling communication accordingly. To help solve these problems, we created a testbed to simulate ambient temperature gradients.

As storage of the harvested energy is also from major importance, we developed a test platform for experiments with energy storage devices [3]. It consists of eight measurement channels for CR2032 coin cells, or connectable storage devices using the pin headers, such as super capacitors. The platform allows to charge or discharge the storage device with pre-defined load patterns. As several papers have shown, batteries are complex systems which are influenced by several factors such as, but not limited to temperature, peak current and duty cycles [6], [1], [2]. This can be used to optimise sleep schedules and duty cycles to improve residual energy.

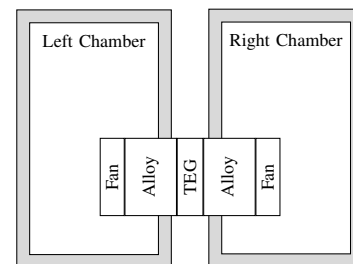


Fig. 1. Setup of the experiment: Two temperature-controller chambers are used to create a temperature gradient

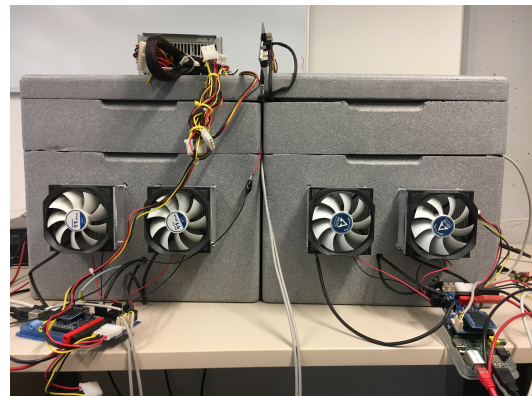


Fig. 2. Image of the setup

II. DEMONSTRATION CONTENTS

Our demonstration will show the exemplary testbed setup as shown in Figure 2. Two of our remote-controllable, temperature-controlled chambers are used to generate a temperature gradient between them. The chambers used in this experiment are an advanced version [4] from our original version [7]. A Thermo-Electric Generator (TEG) is placed between the two chambers. To make the chambers reusable for other experiments, two extension rings are mounted on top, which are replaceable in other experiments in the future. The wall of the chambers and extension rings are 4 cm thick each. A solid block of aluminium is placed in the wall to transfer heat between the TEG and the chamber. Finally, to

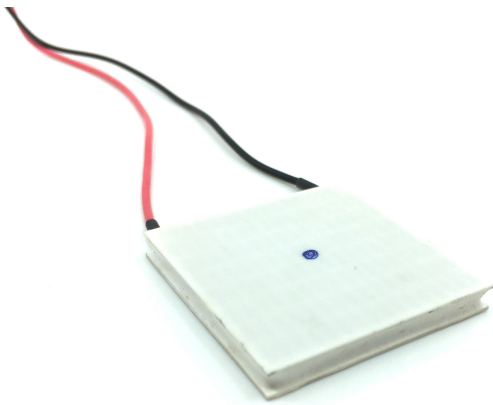
Fig. 3. BK Precision 8600²

Fig. 4. Image of the thermoelectric generator

have a higher temperature difference, two fans are mounted within each chamber and tightened against each other. To measure the temperatures directly at the TEG, two DS18B20 temperature sensors are placed in drilled holes near the edge of the aluminum blocks, near the TEG. The full setup is shown in Figure 2.

An electronic load is either controlled manually or fully automatic by our testbed framework [4]. The BK Precision 8600 is used, as this device can be controlled remotely. We use it in constant resistance mode for maximum power point tracking (MPPT) to derive a simple model of the maximum harvestable power from the TEG. In our experiment the TEG used is a PKE 127 A 0020¹. The thermal conductance is given as 320 mW/K and the thermopower is 51 mV/K. The specified resistance of the element is 3 Ω . An image of the TEG is shown in Figure 4.

III. RESULTS

The presented testbed has recently been used for a feasibility study on energy harvesting from soil temperature [5]. We have recorded temperatures from air and different depths of soil

over the period of one year. Our study analyses the feasibility of harvesting energy from the temperature gradient between air and soil.

Besides results from the study, we have experienced some issues while building the testbed:

(1) The achievable temperature gradient at the TEG is approximately 9° less than the temperature of the chambers. This is expected, as the heat does not transfer perfectly between the block of aluminium and the chamber.

(2) Controlling the temperature precisely at the element throughout the experiment was quite hard, as it is directly influenced by the load. Whenever the load is changed, more or less heat is transferred, which makes it harder to control the temperature gradient at the TEG.

(3) Throughout the experiment we experienced minor errors with the remote control of the electronic load. Improvements to the control script and improved error handling by introducing timeouts and re-transmissions solved the problems.

(4) By generating energy from the heat difference in the chambers, the TEG generates a heat bridge between the chambers, reducing the gradient proportional to the load resistance. As a result, the heating and cooling performance of the chambers is reduced, compared to the operation without the TEG. In our experiments we experienced, that the reduced cooling performance resulted in an increased test duration, because more time was required to reach the desired temperature.

IV. ACKNOWLEDGEMENTS

This paper is part of the REAP project and the research was partially funded by the German Research Council (DFG) under the grant no. BU 3282/2-1.

REFERENCES

- [1] FEENEY, L. M., ROHNER, C., GUNNINGBERG, P., LINDGREN, A., AND ANDERSSON, L. How do the dynamics of battery discharge affect sensor lifetime? In *11th IEEE/IFIP Conf on Wireless On-demand Network Systems and Services (WONS)* (2014).
- [2] FURSET, K., AND HOFFMAN, P. *High pulse drain impact on CR2032 coin cell battery capacity*. Nordic Semiconductor and Energizer, Sept 2011. Technical memo.
- [3] HARTUNG, R., FEENEY, L., ROHNER, C., KÄBERICH, J., WOLF, L., AND GUNNINGBERG, P. A platform for experiments with energy storage devices for low-power wireless networks. In *Proceedings of the 12th ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization* (New Delhi, India, November 2018), WiNTECH '18, ACM. Accepted for publication.
- [4] HARTUNG, R., LICHTBLAU, N., KULAU, U., AND WOLF, L. C. A flexible software framework for real-world experiments and temperature-controlled testbeds. In *Proceedings of the 12th ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation and Characterization* (New Delhi, India, November 2018), WiNTECH '18, ACM. Accepted for publication.
- [5] PULLWITT, S., KULAU, U., HARTUNG, R., AND WOLF, L. C. A feasibility study on energy harvesting from soil temperature differences. In *RealWSN 2018* (Shenzen, China, 2018), ACM. Accepted for publication.
- [6] RAO, R., VRUDHULA, S., AND RAKHMATOV, D. Battery modeling for energy aware system design. *IEEE Computer* 36, 12 (2003).
- [7] VON ZENGEN, G., HARTUNG, R., KULAU, U., BÜSCHING, F., AND WOLF, L. Low cost temperature controlled testbed for wsns.

¹<http://www.peltier4you.de/element.html>

²Image from bkprecision.com

Demo: Scheduling for Passive Undervolting of Peripheral Components

Ulf Kulau, Stephan Friedrichs and Lars Wolf

Technische Universität Braunschweig

Institute of Operating Systems and Computer Networks (IBR)

Email: [kulau|sfriedr|wolf]@ibr.cs.tu-bs.de

Abstract—While existing work focuses on the transceiver and the processing unit to increase the energy efficiency of wireless sensor nodes, it is missed that peripheral energy consumption may dominate that of the entire node. Related to Dynamic Voltage Scaling (DVS), even peripherals’ energy efficiency benefit from a downscaled voltage level, but different peripherals require different minimum voltage levels. With this demo, that has been previously presented at EWSN 2015 [1], we combine theory and practice to present the implementation of an algorithm weighing off the benefits of a downscaled voltage level against the switching overhead.

I. INTRODUCTION

As the dynamic power consumption of CMOS gates shows a quadratic dependency on the voltage level, DVS helps to significantly improve the energy efficiency of microelectronic systems [2]. Hence, several existing DVS approaches [3], [4], [5] lead to an increase of WSN lifetime. Nevertheless, not only MCUs but also peripherals like memory devices, sensors, or actuators benefit from a downscaled voltage level.

Each peripheral hardware device requires a minimum voltage to be properly operated. The common practice is to statically configure the lowest peripheral voltage conform to all peripheral devices’ voltage requirements. This can be very inefficient, because most hardware consumes more energy when exposed to higher voltage. Hence, we seek to exploit a sensor node’s mechanism to dynamically switch the peripheral voltage.

The crux is that switching the voltage does not come for free. If it would, one could simply operate every peripheral device with its minimum required voltage. But switching the voltage consumes energy as well: The additional time interfacing a scalable voltage supply prolongs the duty-cycle of a processing unit, leading to a higher energy consumption.

The concept of incorporating switching cost among power different radio modes was discussed in [6], but is focussed on the radio transceiver only. In the following we outline the algorithm introduced in [7] and demonstrated at EWSN 2015 [1]. This algorithm allows for peripheral voltage scheduling that weighs off the energetic benefits of switching to a lower peripheral voltage against the switching overhead without violating the minimum voltage requirements of active hardware.

Consider a sensor node with a set S of peripheral hardware devices. In order to assess the benefits of switching to a lower peripheral voltage before using $s \in S$, we need to know how much energy is consumed when querying s using the peripheral voltage v . $e_s(v)$ depends on the time t_s necessary to query s , the peripheral voltage v , and the accumulated current

$I_s(v, t)$ flowing through s as well as through the inactive peripheral hardware $S \setminus \{s\}$:

$$e_s(v) = v \int_0^{t_s} I_s(v, t) dt \quad (1)$$

Each $s \in S$ has two attributes: 1. a minimum voltage $v_{\min}(s)$ required to properly operate s , and 2. the energy consumption $e_s(v)$ of all peripherals while only s is active, depending on the peripheral voltage v , see above. Throughout this work, we assume $e_s(v)$ to be a monotonically increasing function, i. e., that a reduction of the peripheral voltage never results in an increased energy consumption. For a constant amount C of energy, the *switching overhead*, the sensor node can adapt its peripheral voltage. The sensor node is presented a sequence of queries denoted by $[1, \dots, n]$, so that the energy consumption E of a voltage schedule is given by:

$$E = \sum_{i=1}^n e_{s_i}(v(i)) + \sum_{i=2}^n \begin{cases} C & \text{if } v(i-1) \neq v(i), \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Our goal is to minimize E , so we call a voltage schedule *optimal* if E is minimal. It follows from the monotonicity of $e_s(v)$ that an optimal schedule only uses $v(i) \in \{v_{\min}(s) \mid s \in S\} = \{V_1, \dots, V_m\}$ with $V_1 < \dots < V_m$.

II. ALGORITHM

Let us, for a pair of query and voltage (i, V_j) , determine the minimum amount of energy $E_{i,j}$ necessary to reach it using a feasible schedule $v(1), \dots, v(i)$ while assuming an infinite energy consumption for infeasible configurations. For the first query, we have:

$$E_{1,j} = \begin{cases} \infty & \text{if } V_j < v_{\min}(s_1), \\ e_{s_1}(V_j) & \text{otherwise.} \end{cases} \quad (3)$$

For $2 \leq i \leq n$, there is the mandatory energy consumption $e_{s_i}(V_j)$ to answer the query i itself, as well as the accumulated costs for traversing $i-1$ preceding configurations. There are two ways to reach the configuration (i, V_j) with an optimal energy consumption: Either the peripheral voltage from the previous query is kept, or it is changed. The former case yields an additional energy consumption of $E_{i-1,j}$. In the latter case we require the minimum amount of energy \hat{E}_{i-1} to reach the cheapest feasible predecessor configuration and the additional costs C for switching the voltage, where $\hat{E}_{i-1} = E_{i-1,\hat{j}}$ with $\hat{j} := \arg \min_j E_{i-1,j}$. This yields, for $2 \leq i \leq n$:

$$E_{i,j} = \begin{cases} \infty & (i, V_j) \text{ is infeasible,} \\ e_{s_i}(V_j) + E_{i-1,j} & E_{i-1,j} < \hat{E}_{i-1} + C, \\ e_{s_i}(V_j) + \hat{E}_{i-1} + C & \text{otherwise.} \end{cases} \quad (4)$$

We use dynamic programming to efficiently solve the recursion by determining $E_{i,j}$ before $E_{i+1,j}$; the optimal overall schedule is that ending in the configuration (n, V_j) , where $E_{n,j} = \hat{E}_n$ is minimal. Our method guarantees that active peripherals never are undervolted, but relies on undervolting the inactive ones (passive undervolting). The general feasibility of passive undervolting has been shown in [8] and we also never observed any issue. However, should a negative effect due to passive undervolting occur, our algorithm could be easily adjusted. For a detailed description of the algorithm and some extensions please refer to [7].

III. IMPLEMENTATION

We use an 8-bit Atmel ATmega1284p MCU as processing unit. Thus, the low computational capabilities of this MCU demonstrates that our approach is sufficiently lightweight for WSN requirements.

Figure 1 shows a picture and a block diagram of the prototype. Compared to ordinary sensor nodes a voltage scaling module is connected to the processing unit via I2C-bus. This module provides a voltage level of $1.8\text{V} \leq v \leq 3.3\text{V}$ with an 8-bit resolution to the peripherals. In this case, the overhead of switching to an arbitrary voltage level is $C \approx 7.76\mu\text{J}$. This includes the increased active time of the MCU and the voltage scaling module's static power dissipation, refer to [4], [7] for details.

Our prototype's sensing unit is divided into an analog and a digital section. The analog section offers the ability of connecting fully analog sensors to the ADC channels of the ATmega1284p, while the digital section includes the devices of Table I. All of them are connected via I2C bus. In order to

Table I. EQUIPPED PERIPHERALS FOR DEMONSTRATION.

Peripheral s	Device	Description	$v_{\min}(s)$ [V]
A	ADXL345	Accelerometer	2.000
E	AT24C08C	EEPROM	1.800
P	BMP085	Pressure Sensor	1.800
G	L3G4200D	Gyroscope	2.400
M	MAG3110	Magnetometer	1.950

calculate an optimal schedule, we need information describing the overall peripheral energy consumption. For this reason, we added a tiny co-MCU to the prototype, which is able to concurrently sample the current consumption of the peripherals (a shunt is used in connection with current sense amplifiers) and to measure the time (the co-MCU can be triggered by the ATmega1284p via digital GPIOs). Hence, $e_s(v)$ can be measured for any given values of s and v .

IV. EVALUATION

Although the demonstration will give the user already the chance to optimize custom schedules, the following table depicts some exemplary schedules to show the general benefit of our approach. The energy savings are compared against three classical strategies. CONSTDEFAULT is what happens when a sensor node has no mechanism to adapt the peripheral voltage. A constant peripheral voltage of 3.3V is kept. CONSTMAXMIN is the trivial strategy that uses the maximum minimum voltage, i. e., $\max_{s \in S} v_{\min}(s)$, for all queries. ALWAYS SWITCH always switches the voltage to its minimum requirement. It ignores the switching overhead.

Table II. SAMPLE SCHEDULES TO SHOW THE BENEFIT OF PERIPHERALS' VOLTAGE SCHEDULING COMPARED TO NAIVE APPROACHES.

Query Sequence	Energy saved by SCHEDULED compared to		
	CONSTDEFAULT	CONSTMAXMIN	ALWAYS SWITCH
AEPGMAEPM	45.80 %	17.13 %	0.97 %
GAMGAMGAM	46.15 %	17.04 %	0.49 %
GAMPE	46.91 %	18.52 %	1.40 %
GPMPGPMP	31.54 %	0.00 %	20.29 %
PAMPE	47.90 %	20.41 %	2.53 %

V. DEMONSTRATION

With a GUI a custom query of peripherals (cf. Table I) can be created. This query is transferred to the prototype board via USB. As the implementation follows a fully self-optimizing approach, the prototype board firstly self-parametrizes the energy functions $e_s(v)$ of involved peripherals. Afterwards, the board executes the optimization algorithm to get the optimal voltage schedule for the given query. Finally the query is processed while the optimal schedule is compared against the trivial voltage strategies as described in the previous section. For this purpose, the second tiny MCU samples the current consumption of CONSTDEFAULT, CONSTMAXMIN, ALWAYS SWITCH and of course SCHEDULED. The results are send back to the PC, where the GUI displays an oscilloscope of the current consumptions as well as an analysis of the saved energy.

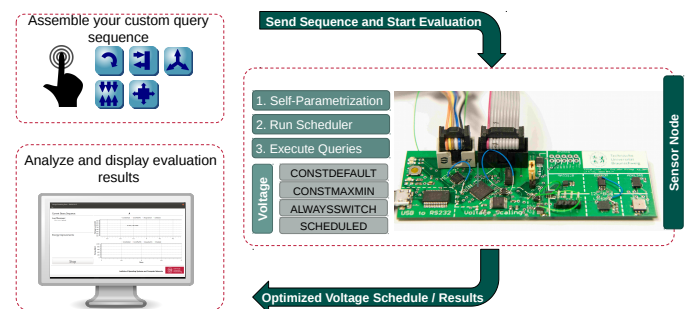


Figure 1. Picture of the implementation and illustration of the actual demo.

REFERENCES

- [1] U. Kulau *et al.*, "Demo abstract: Voltage scheduling of peripheral components on wireless sensor nodes," *ewsn 2015*, 2015.
- [2] U. Tietze and C. Schenk, *Electronic Circuits: Handbook for Design and Application*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.
- [3] L. Hoermann *et al.*, "Energy Efficient Supply of WSN Nodes using Component-Aware Dynamic Voltage Scaling," *11th European Wireless Conference 2011 - Sustainable Wireless Technologies*, 2011.
- [4] U. Kulau *et al.*, "A node's life: Increasing WSN lifetime by dynamic voltage scaling," in *The 9th IEEE International Conference on Distributed Computing in Sensor Systems 2013 (IEEE DCoSS 2013)*, 2013.
- [5] —, "Idealvoting: Reliable undervolting on wireless sensor nodes," *ACM Transactions on Sensor Networks (TOSN)*, vol. 12, no. 2, p. 11, 2016.
- [6] R. Jurdak *et al.*, "Radio sleep mode optimization in wireless sensor networks," *IEEE Transactions on Mobile Computing*, 2010.
- [7] S. Friedrichs *et al.*, "Energy-efficient voltage scheduling of peripheral components on wireless sensor nodes," in *Communications Workshops (ICC), 2014 IEEE International Conference on*, 2014.
- [8] H. Jayakumar *et al.*, "Hypnos: an ultra-low power sleep mode with sram data retention for embedded microcontrollers," in *Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis*. ACM, 2014.