Dependable wireless sensor networks for in-vehicle applications

Dependable Wireless Sensor Networks for In-Vehicle Applications

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Technische Universiteit Eindhoven, op gezag van de
rector magnificus prof.dr.ir. F.P.T. Baaijens, voor een
commissie aangewezen door het College voor
Promoties, in het openbaar te verdedigen op
woensdag 12 december 2018 om 16:00 uur

door

Rasool Tavakoli Najafabadi

geboren te Najafabad, Iran

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

voorzitter:	prof.dr.ir. J.H. Blom
1^e promotor:	prof.dr K.G.W. Goossens
2^e promotor:	prof.dr.ir T. Basten
co-promotor:	dr. M. Nabi Najafabadi
leden:	prof.dr. K.G. Langendoen (Technische Universiteit Delft)
	prof.dr.ir. S.M. Heemstra de Groot
	prof.dr. J.J. Lukkien
	dr.ir. H.G.H. Vermeulen (NXP Semiconductors)

Het onderzoek of ontwerp dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

Dependable Wireless Sensor Networks for In-Vehicle Applications

Rasool Tavakoli Najafabadi

Doctorate committee:

prof.dr K.G.W. Goossens	Eindhoven University of Technology, first promotor
prof.dr.ir T. Basten	Eindhoven University of Technology, second promotor
dr. M. Nabi Najafabadi	Eindhoven University of Technology, copromotor
prof.dr.ir. J.H. Blom	Eindhoven University of Technology, chairman
prof.dr. K.G. Langendoen	Delft University of Technology
prof.dr.ir. S.M. Heemstra de Groot	Eindhoven University of Technology
prof.dr. J.J. Lukkien	Eindhoven University of Technology
dr.ir. H.G.H. Vermeulen	NXP Semiconductors

This work was supported in part by the ARTEMIS Joint Undertaking, project DEWI.

Cover design by Rasool Tavakoli Najafabadi

Printed by: Gildeprint - The Netherlands

A catalogue record is available from the Eindhoven University of Technology Library
ISBN: 978-90-386-4645-9

This thesis is dedicated to my best friend, Marzie.

Abstract

Dependable Wireless Sensor Networks for In-Vehicle Applications

In new generations of vehicles, a considerable percentage of the vehicle weight is due to all cables required for control and data communications between different parts of the vehicle. Replacing parts of wiring of In-Vehicle Networks (IVNs) by wireless communications is a promising approach to reduce installation costs and increase the flexibility and reconfigurability of the system. Based on the IVN requirements, the most important bottleneck towards realizing in-vehicle Wireless Sensor Networks (WSNs) is dependability of wireless communications. This includes the availability, reliability, and maintainability of the system over time. Exploiting general-purpose protocol stacks in WSNs is sub-optimal for this type of applications. The main differences of IVNs compared to other networks are the frequently changing wireless noise conditions, high density of wireless sensor nodes, and a wide range of sensor types with different capabilities and application requirements. Energy consumption of nodes should be minimized to extend the lifetime of battery-powered sensors within the network.

The main concern of this thesis is to *provide a dependable protocol stack for low-power in-vehicle WSNs*. The Time-Slotted Channel Hopping (TSCH) mode of the low-power IEEE 802.15.4 standard protocol is considered as the Medium Access Control (MAC) layer for this purpose. TSCH improves reliability of the IEEE 802.15.4 standard protocol by using guaranteed medium accesses and channel diversity. We address dependability challenges of in-vehicle WSNs at three points in the protocol stack; i.e., dynamic channel conditions at physical and MAC layers, high density of nodes at the network layer, and traffic heterogeneity of in-vehicle WSNs at the MAC layer.

Cross-technology interference on the license-free ISM bands is one of the main factors that affects the performance of WSNs. Using real-world experiments, we studied the cross-technology interference behavior in in-vehicle environments under different scenarios. The results show that the distribution of interference on different IEEE 802.15.4 channels is typically non-uniform. Moreover, interference on each channel is changing over time.

The dynamic interference in in-vehicle WSNs leads to non-guaranteed reliability of the communications over time. We propose an Enhanced version of the TSCH protocol (ETSCH) that dynamically detects good quality channels to be used for communication. The quality of channels is extracted using a combination of a central and a distributed channel-quality estimation technique. These techniques enable ETSCH to follow dynamic interference without any negative effect on the throughput of the network. Experimental and simulation results show that ETSCH improves reliability of network communications by reducing communication failures and the maximum length

of burst packet losses, compared to basic TSCH and the state-of-the-art solutions.

Satisfying the stringent requirements of in-vehicle networks is challenging and demands for special consideration in network formation and TSCH scheduling. Accordingly, a cross-layer Low-Latency Topology management and TSCH scheduling (LLTT) technique is proposed that provides a very high timeslot utilization for the TSCH schedule and minimizes communication latency. It first picks a tree-topology structure for the network that increases the potential for parallel TSCH communications. Then, by using an optimized graph isomorphism algorithm, it extracts a proper network topology for the selected tree structure in the physical connectivity graph of the network. This network topology is used by a light-weight TSCH schedule generator to provide low data-delivery latency. Two techniques, namely grouped retransmission and periodic aggregation, are exploited to increase the performance of the TSCH communications. The experimental results show that LLTT reduces the end-to-end communication latency compared to other approaches, while keeping the communications reliable by using dedicated links and grouped retransmissions.

The TSCH protocol defines two types of timeslots for communications, namely dedicated and shared timeslots. A TSCH scheduler uses these timeslots to design a communication schedule for the network links, based on the required bandwidth for each link. In-vehicle WSNs connect several types of sensors in a vehicle, each type running a different application with different modes of operation over time. This leads to time-varying data traffic generation by each node and poor efficiency of static TSCH schedules. We propose a new type of timeslot, called hybrid timeslot, to support this time-varying behavior of in-vehicle WSNs. A hybrid timeslot acts as a dedicated timeslot for a specific link, when there are packets available to be transmitted on that link. Otherwise, it acts as a shared timeslot that can be accessed by other links, using a contention-based mechanism. Experimental and simulation results show that using hybrid timeslots instead of dedicated timeslots in a TSCH schedule reduces the communication delay by half on average, while keeping the communications reliable.

We demonstrate the applicability of the presented techniques using real-world experiments with IEEE 802.15.4-enabled devices. Different setups and configurations are used for evaluations. We show that our techniques are implementable and improve the dependability of WSNs for in-vehicle applications.

Contents

1

Introduction

1.1 Wireless Sensor Networking

Wireless Sensor Networks (WSNs) are networks composed of tiny wireless-enabled devices, with low processing facilities, distributed over a platform. Each device is called a *sensor node* or *wireless node* and can be embedded in various objects to sense their condition or generate actuation signals. Wireless nodes communicate to each other to form a networked embedded system to monitor and control the target platform.

WSNs cover a wide range of applications from smart buildings to smart vehicles. They eliminate the use of unnecessary wires to connect sensors and actuators to the central unit of the platform. Other than reducing the wiring costs of the system, use of wireless sensors reduces the production and maintenance costs. Moreover, it gives the sensors and actuators the flexibility to be installed on the moving parts of the targeting platform. Considering a network that is deployed in a vehicle, as an example, tire pressure sensors with wireless communication capability can be easily installed inside the tires.

Wireless nodes are usually considered to be stand-alone. Thus, it is expected to use small batteries or an energy harvesting technique as power supply for them. Changing batteries is very costly or even not possible in some applications. This demands for design of special platforms and communication protocols for wireless nodes to keep them operational for years by one small battery. On the other hand, wireless nodes are going to replace wired sensors that are usually very low cost and this replacement should be economical. Otherwise, there is no interest from the targeting industry to use WSNs. These requirements limit both computation and communication capabilities of wireless nodes. And, regardless of these limitations, applications have their own requirements. These requirements are usually expressed by a number of Quality-of-Service (QoS) constraints that should be met by the WSNs. Some of the QoS requirements are throughput, latency, and maximum tolerable data misses. The designer of the WSN should consider network limitations and application requirements together in the design process.

The IEEE 802.15.4 standard [2] was introduced in 2003 to address the limitations

of WSNs. It defines physical and Medium Access Control (MAC) layers for low-power and low-rate wireless networks. Currently, the IEEE 802.15.4-enabled wireless sensors are widely used in different applications, and cost only a few dollars per sensor node. In 2015, a new version of this standard was introduced with a number of new MAC techniques to increase the reliability of communications in industrial applications. Time-slotted communications and channel hopping are two important concepts that are presented in this version. There is a lot of research on the design of dependable WSNs based on the IEEE 802.15.4 protocol, targeting different application domains. In general, higher dependability enables more applications to use wireless communications as a replacement of wired communications.

1.2 In-Vehicle Applications

Wireless communication is considered as a solution to be used in new generations of modern vehicles. This technology provides significant improvement in flexibility and reconfigurability of In-Vehicle Networks (IVNs) to reduce installation and maintenance costs. Moreover, replacing parts of wired IVNs by wireless communications reduces the vehicle weight and accordingly its fuel consumption.

In an IVN, there are different types of wired sensors/actuators placed in different parts of the vehicle, from engine to tires and lights, that could be replaced with wireless sensors/actuators. Figure 1.1 shows an example IVN in a truck, consisting of wired and wireless connections. The targeting applications to be supported by an automotive WSN are considered to be non-safety related. A study by Volvo truck company shows that a high end truck can have around 150 sensors, and assuming that at least 20% of this number can be migrated to short range wireless links, we would have an automotive WSN with a node population of around 30 sensor nodes [57]. This replacement is possible if the QoS of the provided communication by the WSN meets the related application requirements. Therefore, by providing a higher level of QoS by the WSN, more wired sensors/actuators can be potentially replaced with the wireless sensors/actuators. Some of the targeting sensors are tire-pressure sensors, fuel-level sensors, window buttons and actuators, parking sensors, actuators of head and tail lights, actuators of indicator lights, seatbelt sensors, non-safety engine sensors, and rain sensors.

While the concept of WSNs is well defined and a wide range of technical solutions are available for them, there is a lack of clarity in the technical solutions for the automotive domain. This is because of the unique characteristics of automotive WSNs that makes them different from typical WSNs. Some of these characteristics are as follows.

Small area and high node density: An automotive WSN is limited to the area of a vehicle that can be only few square meters in a compact car and at most the size of a truck. In this limited area, a node population in the order of tens to a hundred of nodes is expected. This leads to a node density that is much higher than the typical WSNs that are used for many applications such as environmental monitoring. As wireless sensors and their positions are predefined in an automotive WSN, there is no need for network expandability and the network density stays the same during its lifetime.

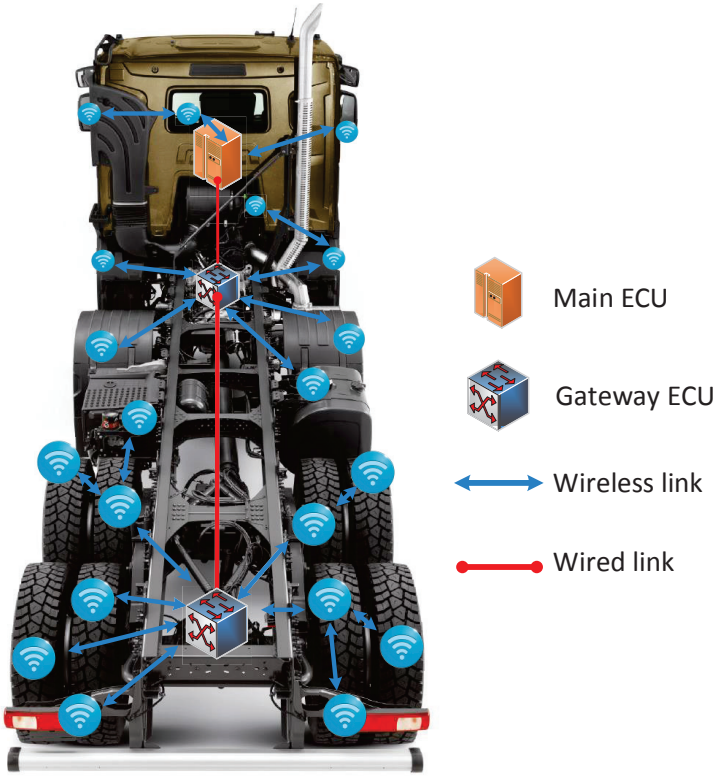


Figure 1.1: An example IVN in a truck, consisting of wired and wireless connections (*copyright Renault Trucks SASU*).

Simple network topology: The data traffic in an automotive WSN typically ends up at a coordinator Electronic Control Unit (ECU) or is sourced from it. Considering a static topology to be used for the network communications, a tree topology is one of the good candidates that can be used. Considering the size of the automotive WSNs and the typical communication range of the IEEE 802.15.4, which is easily a few meters, the maximum number of hops to have a fully connected network is one or two.

Spatial heterogeneity: In an automotive WSN, the distribution of sensors/actuators over a vehicle is non-uniform and the density of nodes may be higher in some parts (e.g., engine area), causing local bandwidth overloading. Moreover, the wireless link quality may differ in different parts of a vehicle, caused by obstacles (e.g., vehicle body or passengers) and/or wireless interferers (e.g., WiFi and Bluetooth enabled devices inside a vehicle). This divides the network into different geometric modules that each present unique operational conditions. Moreover, automotive networks connect several types of sensors and actuators in a vehicle to a central entity, each type running a different application with different bandwidth requirements. Thus, the data communication traffic is quite diverse in different parts of the vehicle. These variations require adaptations that should be properly supported by the WSN to prevent local failures.

Temporal heterogeneity: A WSN can be in different operation modes based on the vehicle status; e.g., sleep, normal, high speed, and emergency. Depending on the operation mode of the vehicle, the communication rate of the periodic applications may change, which causes traffic diversity. Moreover, some sensors in an automotive WSN generate event-based data and impose a level of unpredictability to the bandwidth demand. This temporal heterogeneity is mainly caused by the application layer variations.

The physical layer of the IEEE 802.15.4 standard [7] defines 16 channels in the unlicensed 2.4 GHz ISM band. Other protocols such as Bluetooth and Wi-Fi also use the same frequency band for their communications. As a vehicle is driven along the city roads or highways, the automotive WSN in it experiences different levels of wireless interference over time. This is caused by the wireless devices that are operating along the roads or are inside other vehicles. Moreover, passengers of a vehicle may use wireless-enabled devices that cause strong interference for the automotive WSN. Depending on the number of coexisted wireless devices and their distance, the interference conditions may change a lot during movement of a vehicle. On the other hand, due to the metallic body of the vehicle, the in-vehicle wireless communications experience stronger attenuation, shadowing, and multipath fading that may change over time. This variable wireless medium at the physical layer leads to temporal heterogeneity of automotive WSNs. This should be properly supported by the wireless network protocol stack. Otherwise, it is not possible to guarantee the required QoS for the user applications.

Heterogeneity in power source: In general, a wireless node could be battery-powered, energy harvesting-powered, or wire-powered. Wire-powered sensors are rarely used in WSNs, but actuator nodes could be wire-powered, as they need the power line anyway (e.g., all parts in the car that do electro-mechanical tasks). In this case, they get the control signal through a wireless link. Battery-powered and energy harvesting-powered wireless nodes usually have limited functionality due to the power constraints. On the other hand, wire-powered sensor nodes can be more functional and handle more communication tasks in the network.

1.3 Dependable Networking

In this thesis we aim to provide a *dependable* WSN to partly replace wired communications in vehicles. Dependability of a system is defined as the ability of the system to avoid service failures that are more frequent than is acceptable [11]. It is expressed by a number of quantitative and qualitative measures. *Reliability* and *availability* are considered as the quantifiable measures of dependability of a system. Qualitative measures, such as safety and maintainability, are more subjective and cannot be measured directly with metrics. In this thesis, we focus on the quantitative measures of dependability to extract the performance of the automotive WSNs under test.

Reliability of a system shows the continuity of its correct service, while availability addresses the readiness of the system for providing correct service [11]. The correct service of a WSN is defined by a number of QoS requirements that should be met. Considering the unstable wireless communication conditions of automotive WSNs and disturbances, the provided service by the WSN may experience failures over time. Thus,

the ability of the WSN to handle these disturbances and satisfy QoS requirements under various conditions over time affects both reliability and availability of the WSN. An improved QoS guarantees a more dependable WSN in terms of reliability and availability. Accordingly, dependability requirements should be determined by the user applications as some QoS requirements so that the designer of the WSN can prove (through analysis or testing) the dependability of the WSN.

The QoS requirements of a system are defined by a number of performance metrics. Some of the performance metrics that are typically used for evaluation of WSNs are communication success ratio, end-to-end latency, length of burst packet losses, throughput, and energy consumption. When the provided performance is at least the defined QoS, the service of the WSN is considered correct.

An IVN is typically used to transfer data from sensors to the main ECU of the vehicle or from the ECU to actuators. Automotive WSNs are considered to be used for non-safety applications of IVNs. Since many sensors and actuators interact with the driver or passengers on board, they have a set of QoS requirements to keep the user experience at an acceptable level. Usually, the user does not care what the provided end-to-end latency is, or what is the communication success ratio. Instead, some requirements are defined at a higher level of abstraction to guarantee the needed functionality for the user. For instance, the delay between pressing a window button and moving the window should be less than 200 ms to be hidden from the user [9]. These higher level requirements are converted to QoS requirements for different performance metrics.

From the manufacturing point of view, wireless nodes in an automotive WSN are expected to be low-cost and some of them should operate with small batteries. Accordingly, there are some limitations for their functionality. The main limitation is the low-rate communication nature of the wireless protocols that are defined for WSNs. This makes the network throughput limited and makes it very challenging to guarantee the timing requirements of real-time applications. In general, timing requirements of real-time applications are of two types; hard and soft real-time requirements. Hard real-time requirements are deadlines that should not be missed; otherwise the functionality of the application is incorrect and it faces a failure. This type of requirements is usually safety related and is considered to only use highly reliable wired communications. On the other hand, infrequent misses of soft real-time requirements are tolerable, but the usefulness of a result degrades after its deadline. This degrades the provided QoS for the application. *In this thesis, we use real-time to refer to soft real-time requirements, since wireless communications are not fully reliable and cannot be used for safety-critical applications.*

Considering stochastic behavior of automotive WSNs and soft real-time applications that are considered here, extracting performance for the worst-case requirements may not be necessary for requirement verification. This is because worst-case scenarios usually happen infrequently and infrequent misses of requirements are tolerable. Instead, the average-case performance can be used for verification. This leads to best-effort system design to maximize the average-case performance. In this case, the performance distribution is an important factor for system QoS verifications.

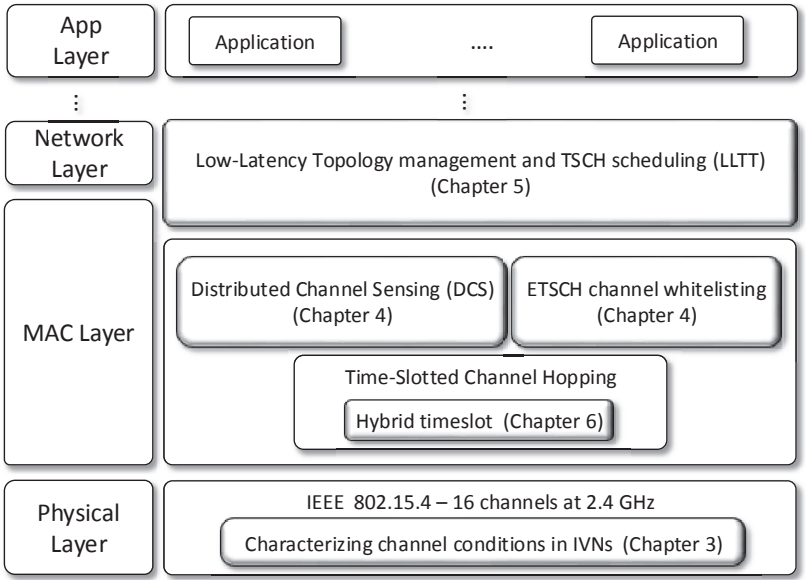


Figure 1.2: Overview of chapters and their positioning within networking protocol stack.

1.4 Problem Statement and Contributions

The high-level research question in this thesis is: *How should a WSN protocol stack be designed so that it 1) handles communication uncertainties caused by wireless interference and provides dependable communication for real-time automotive applications, 2) takes power constraints of the sensor nodes into consideration, and 3) supports heterogeneity requirements at different layers.*

This thesis proposes different mechanisms in different networking layers to provide a dependable solution for automotive WSN. Figure 1.2 shows the positioning of the proposed mechanisms within the networking protocol stack. Table 1.1 gives an overview of the challenges in state-of-the-art IVNs addressed by the proposed solutions. Considering the requirements of the automotive WSNs, the Time-Slotted Channel Hopping (TSCH) mode of the IEEE 802.15.4 standard [7] is selected as the MAC layer protocol for these networks. TSCH reduces the effects of interference and multi-path fading through guaranteed medium access and channel diversity. It improves the reliability of low-power wireless communications through the guaranteed access. Accordingly, different mechanisms are proposed on top of the TSCH protocol to increase the reliability (and availability) of automotive WSNs. Performance of the proposed techniques is studied through real-world experiments. The experiments are complemented with computer simulations to evaluate the proposed techniques in a wider range of configurations.

The MAC layer is responsible of controlling the physical layer accesses of radio transmitters in the network to the medium. The IEEE 802.15.4 protocol defines 27 channels in different frequency bands. 16 channels of these 27 channels, which are defined in the 2.4 GHz unlicensed ISM band, provide higher bandwidth compared to the other

Table 1.1: An overview of some existing problems in automotive WSNs and the proposed solution for each one of them in this thesis.

Challenge	Underlying reason	Technique
Unknown interference behavior in automotive WSNs	- Movement of vehicles - Wireless devices in vehicles	Real-world measurements
Unstable performance of automotive WSNs	Dynamic interference	ETSCH+DCS
High-latency convergecast in automotive WSNs	- High node density - dependency between links	LLTT
Dynamic bandwidth requirements by each link in automotive WSNs	Traffic heterogeneity	Hybrid timeslots

channels. However, the number of wireless devices that operate in this band is ever increasing. This broad usage of the same RF band may cause considerable performance degradation of WSNs due to cross-technology interference. There is a lot of work done on the coexistence of the IEEE 802.15.4 protocol and other standard technologies such as IEEE 802.11 WLAN [5] and IEEE 802.15.1 Bluetooth [1], but none of them considers the highly dynamic conditions of automotive WSNs. Moreover, as the TSCH MAC enables the use of multiple channels for a communication link through a channel hopping technique, the interference behavior is not only important over time, but also over different channels. Thus, in this thesis, we first investigate the interference behavior in automotive environments using real-world experiments (Chapter 3). We consider different scenarios and measure the interference on all the 16 channels of IEEE 802.15.4 in the 2.4 GHz band. To study the effect of interference on in-vehicle networks, we use this data set to evaluate the performance of a TSCH link. The simulation results show that the packet error rate for some interference scenarios is very dynamic over time. This reveals the importance of using adaptive interference mitigation techniques to improve the reliability of automotive WSNs.

Considering the cross-technology interference effect on automotive WSNs, this thesis proposes an Enhanced version of the TSCH protocol together with a Distributed Channel Sensing technique (ETSCH+DCS) which dynamically detects good quality channels to be used for communication (Chapter 4). The quality of channels is extracted using a combination of a central and a distributed channel-quality estimation technique. The central technique uses the Non-Intrusive Channel-quality Estimation (NICE) technique which proactively performs energy detections in the idle part of each timeslot at the central node of the network. NICE enables ETSCH to follow dynamic interference, while it does not reduce throughput of the network. The distributed channel quality estimation technique is executed by all the nodes in the network, based on their communication history, to detect interference sources that are hidden from the central node. We did two sets of lab experiments with controlled interferers and a number of simulations using real-world interference data sets to evaluate ETSCH. Experimental and simulation results show that ETSCH improves reliability of network communications, compared to basic TSCH and alternative solutions. In some experimental scenarios NICE itself has been able to increase the average packet reception ratio by 22% and to shorten the

length of burst packet losses by half, compared to the plain TSCH protocol. Further experiments show that DCS can reduce the effect of hidden interference (which is not detectable by NICE) on the packet reception ratio of the affected links by 50%.

Automotive WSNs are required to be reliable and have low-latency data delivery, while the network is very dense. However, satisfying the stringent requirements of these networks is challenging and demands for special consideration in network formation and TSCH scheduling. Targeting convergecast (collecting all data from sensors in the central ECU) in dense automotive WSNs, a cross-layer Low-Latency Topology management and TSCH scheduling (LLTT) technique is proposed (Chapter 5). It provides a very high timeslot utilization for the TSCH schedule and minimizes communication latency. It first picks a tree-topology for the network that increases the potential of parallel TSCH communications. Then, by using an optimized graph isomorphism algorithm, it extracts a proper network topology for the selected tree structure in the physical connectivity graph of the network. This network topology is used by a lightweight TSCH schedule generator to provide low data-delivery latency. Two techniques, namely grouped retransmission and periodic aggregation, are exploited to increase the performance. The experimental results show that LLTT reduces the end-to-end communication latency compared to other approaches, while keeping the communications reliable by using dedicated links and grouped retransmissions.

The IEEE 802.15.4 TSCH protocol defines two types of timeslots for communications, namely dedicated and shared timeslots. These timeslots are used to design a communication schedule for the network links, based on the required bandwidth for each link. Considering heterogenous automotive WSNs with time-varying data traffic generation by each node, the bandwidth requirements are changing over time for each link. This leads to poor efficiency of a predefined schedule when there is no data traffic for the dedicated timeslots, or there is too much data traffic injected to the shared timeslots. In this thesis, we propose a new type of timeslot, called hybrid timeslot (Chapter 6). A hybrid timeslot acts as a dedicated timeslot for a specific link, when there are packets available to be transmitted on that link. Otherwise, it acts as a shared timeslot that can be accessed by other links, using a contention-based mechanism. The hybrid timeslot has backward compatibility with the TSCH protocol and is functional with a few adaptations in the parameter setup of the TSCH protocol. Experimental and simulation results show that for heterogeneous automotive WSNs, using hybrid timeslots improves communication latency without a reliability penalty. This enables serving more applications with an automotive WSN, as higher QoS is provided for applications with more restricted real-time requirements.

1.5 Thesis Overview

The remainder of this thesis is structured as follows. Chapter 2 introduces the necessary background of TSCH networks and the terminology that is used throughout the thesis for network architecture and performance evaluation methods. Chapter 3 investigates the interference behavior in in-vehicle environments using real-world experiments. Interference sources are categorized accordingly and by using simulations and the collected interference data sets, the effect of interference on the automotive TSCH WSNs is

studied. Chapter 3 is based on publication [73]. Considering the interference behavior in automotive WSNs, the ETSCH+DCS interference avoidance technique is proposed on top of TSCH in Chapter 4. Chapter 4 is based on two publications [72, 74]. Considering the high density of nodes in an automotive WSN, Chapter 5 introduces the LLTT cross-layer network management and TSCH scheduling technique in a higher networking layer to provide low end-to-end communication delays for the automotive WSNs. Chapter 5 is based on publication [77]. Chapter 6 focuses on the communication heterogeneity within automotive WSNs and describes the hybrid timeslot design that is proposed to handle this. Chapter 6 is based on publication [76]. Chapter 7 concludes the thesis and provides a vision of future research directions in the field of automotive WSNs.

2

Background and Terminology

This chapter describes the terminology that is used in this thesis. It starts with presenting the necessary definitions and assumptions, and continues with introducing the Time-Slotted Channel Hopping (TSCH) Medium Access Control (MAC) protocol. In this thesis, TSCH is used as the MAC layer communication protocol standard for automotive WSNs. Then, the Quality-of-Service (QoS) metrics that are used for the performance evaluations are defined. Finally, the evaluation methods for the QoS metrics are explained.

2.1 Network Model and Definitions

An automotive WSN that is deployed in a vehicle (e.g., a passenger car or truck) consists of a number of nodes that their positions are almost static in the vehicle. Due to the movement of the vehicle parts of which some sensors are a part, positioning of some nodes within the network may occasionally change. For example, when tires are rotating, the position of the tire pressure sensors may change. Also, position of the doors may be temporarily changed, if doors are opened. In this thesis, we consider a static network model in which the small positional variation is captured in link quality fluctuations.

Suppose that $V = \{n_i : 1 \leq i \leq N\}$ is the set of N wireless nodes deployed in an in-vehicle WSN. $L_{i,j} = \{[n_i, n_j] : n_i \in V, n_j \in V\}$ is called a link and defines a unidirectional [source, destination] communication pair in the network. Links are used to construct the network communication topology. A subset of nodes $G \subset V$ contains the *gateways* of the network. Gateways are connected to the main Electronic Control Unit (ECU) of the vehicle using wired connections. They collect the sensory data from the WSN and forward them to the main ECU (Figure 1.1). They also get the actuation commands from the ECU of the car and transfer them to the right wireless nodes that are connected to the actuators. In this thesis, we consider one gateway to be deployed in the network. The proposed techniques can be extended to support networks with multiple gateways (as Figure 1.1) in a straightforward way. We leave this extension as future work.

According to the distance between communicating wireless nodes and their communication range, single or multihop connections may be used in the network. Single-

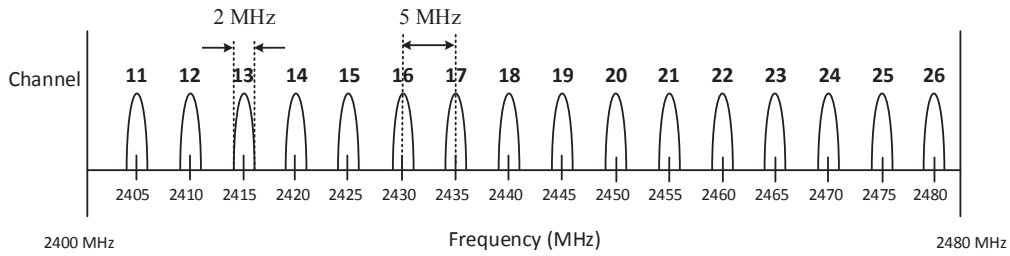


Figure 2.1: IEEE 802.15.4 channels in the 2.4 GHz ISM band.

hop refers to the communications over one link between two nodes. In a multi-hop connection, multiple links are used to deliver the data on a path from a source node to a destination node. Due to the small network dimensions of automotive WSNs, network connections with one or two hops are usually enough to connect all nodes to the gateway(s). Experimental results that are presented in Chapter 5 show that even a star topology with one-hop communications can form an automotive WSN in a passenger car.

2.2 IEEE 802.15.4

The IEEE 802.15.4 standard was introduced in 2003 and specifies the physical (PHY) and MAC layers for Low-Rate Wireless Personal Area Networks (LR-WPANs). A number of known WSN technologies such as ZigBee, WirelessHART [68], and ISA100.11a [3] use this standard as their basis of the networking protocol stack. The most recent version of this standard was published in 2015 [7] which introduces a number of new MAC mechanisms to improve support for industrial applications. TSCH is one of these mechanisms that uses time-slotted communications together with a channel hopping technique to improve the reliability of communications. In this section, we briefly introduce the IEEE 802.15.4 PHY and some of its functionalities. Then, we continue with a short overview of the TSCH MAC mechanism.

2.2.1 Physical Layer

The IEEE 802.15.4 protocol standard edition 2015 [7] defines 19 different PHYs with different combinations of spread spectrum and digital modulations. The Direct Sequence Spread Spectrum (DSSS) PHY employing Offset Quadrature Phase-Shift Keying (O-QPSK) modulation is a widely used PHY for IEEE 802.15.4 compliant devices. This PHY can be used for all the three main frequency bands of 868, 915, and 2450 MHz. It offers a data rate of 250 kb/s in both the 915 and 2450 MHz bands and a data rate of 100 kb/s in the 868 MHz band. The 2450 MHz band (2.4 GHz ISM band) is a worldwide unlicensed spectrum and has been selected as the primary IEEE 802.15.4 band in the 2006 edition of the standard.

This standard uses a combination of channel numbers and channel pages to define channels in different frequency bands. Channel page zero supports 27 channels that are

defined in the 2003 edition of the standard. These channels are numbered from 0 to 26 as channel 0 in the 868 MHz band, channels 1 to 10 in the 915 MHz band, and channels 11 to 26 in the 2.4 GHz band. Channels that are defined in the 2.4 GHz band use 2 MHz receiver bandwidth with a channel spacing of 5 MHz (Figure 2.1).

Besides data transmission, data reception, and channel selection, an IEEE 802.15.4 compliant PHY shall be able to perform a number of other tasks including Energy Detection (ED), Clear Channel Assessment (CCA), and Link Quality Indicator (LQI). An ED is the average of the received signal power within the bandwidth of the channel over 8 symbol periods (128 μs). There is no need to decode or identify the signal on the channel to perform an ED. This value is bounded with a minimum ($ED^{Min} = 0$) and a maximum (ED^{Max}) value (platform dependent) which is linearly mapped to the received power in dB, as described in the IEEE 802.15.4 [7] standard document.

A CCA is an indication of the medium status and is used to check if a channel is busy or idle. Four operation modes are defined to calculate the CCA result:

1. detecting any energy above a predefined threshold,
2. detecting a signal that is compliant with the currently used PHY,
3. a logical combination of the first and second modes with an AND or OR operator,
4. and always reporting an idle medium.

LQI indicates the strength and/or quality of the used link for a packet communication. Accordingly, this measurement is performed for each received packet at the PHY. The LQI value may be calculated based on the receiver ED of the incoming packet, a signal-to-noise ratio estimation, or a combination of these methods. Its value is in the range of 0x00 to 0xff and should be associated with the lowest and highest quality estimation by the receiver.

2.2.2 Time-Slotted Channel Hopping MAC

TSCH is defined as one of the MAC operating modes of the IEEE 802.15.4 standard [7] to support industrial applications. It increases reliability of communications against internal/external interference and multi-path fading. This is done through time-slotted communications with a predefined pattern and also channel hopping.

TSCH divides time into fixed time periods called *timeslots*. The timeslot duration, *macTsTimeslotLength*, is long enough for transmission of a maximum size packet and its Acknowledgement (ACK). TSCH uses device-to-device synchronization to keep all the nodes of a network synchronized. Timeslot synchronization is necessary to establish communication between pairs of nodes. A receiver node should be aware of the start of the sender's timeslot to turn on its radio and listen to the medium before transmission starts. Because of the clock drift between nodes, the synchronization process needs to be continuously performed to keep nodes synchronized.

TSCH defines a diagram for timeslots, shown in Figure 2.2. To compensate an amount of timeslot phase differences caused by clock drifts, this diagram introduces a number of offsets. There is a *Rx* offset (*macTsRxOffset*) at the beginning of a receiver's timeslot before it starts listening to the medium. This *Rx* offset prevents interference

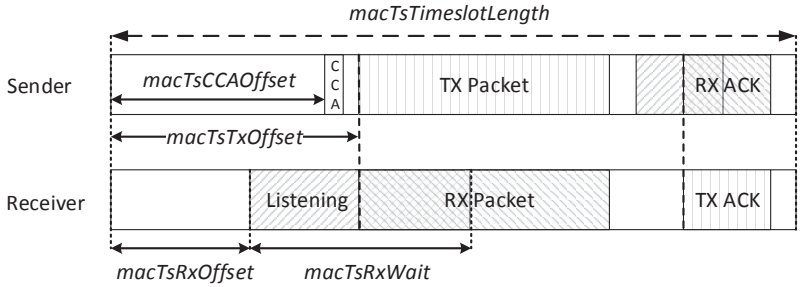


Figure 2.2: The structure of transmit and receive timeslots in IEEE 802.15.4 TSCH mode.

from other nodes in the network which are behind for a maximum time of *macTsRxOffset* and still are transmitting in the previous timeslot. A sender node starts its packet transmission after a *Tx* offset (*macTsTxOffset*) from the beginning of a timeslot. This offset is defined with a value greater than the *Rx* offset to make the communication possible when the sender is ahead of the receiver for a *macTsTxOffset – macTsRxOffset* period of time. A CCA offset (*macTsCCAOffset*) is defined for a sender to perform a CCA before each *Tx* and prevent packet transmission in the case of a busy channel. When a receiver starts to listen to the medium for a packet reception in a timeslot, it waits for a *macTsRxWait* period of time to receive the packet. If the transceiver cannot detect any packet preamble in this period, the receiver stops listening. The values of these parameters are defined in such a way so that *macTsRxOffset + macTsRxWait* is greater than *macTsTxOffset*. Thus, the communication can be successful if the receiver is ahead of the sender for at most the time difference of these two values. Some other timings such as *Tx/Rx* durations and *ACK* transmission timings are defined in the protocol but these are not relevant to this thesis, and thus not shown in Figure 2.2.

$SF = \{Slot_1, ..., Slot_{L_{SF}}\} \rightarrow \mathcal{P}(L)$ is called a *slotframe* and consists of L_{SF} timeslots. Each timeslot is assigned to a subset of L (P(L) denoting the *power set* of L), where L is the set of links in the network. This means that each timeslot can be either idle, dedicated to one link, or be shared between multiple links for communications. A network may use multiple slotframes with different length for communications. However, in this thesis we consider the length of all slotframes in a network to be the same (L_{SF}). Slotframes repeat over time to enable links to have periodic access to the medium. Longer slotframes lead to longer periods for communications of each timeslot.

Each timeslot can be either dedicated to one link for communications or be shared between multiple links. Dedicated timeslots avoid collisions and internal interference. Shared timeslots are assigned to more than one link for transmission. This may lead to collisions that result in a transmission failure. To reduce the probability of collisions in shared timeslots, TSCH uses a slotted Carrier-Sense Multiple Access with Collision Avoidance (CSMA-CA) algorithm for transmissions in these timeslots. In this technique, each wireless node should wait for a random number of shared timeslots in the range 0 to 2^{BE}–1 (backoff window) before transmitting a packet in a shared timeslot. *BE* is the backoff exponent and is increased by 1 for each consecutive failed transmission in a shared timeslot. This reduces the collision probability to access the shared timeslot,

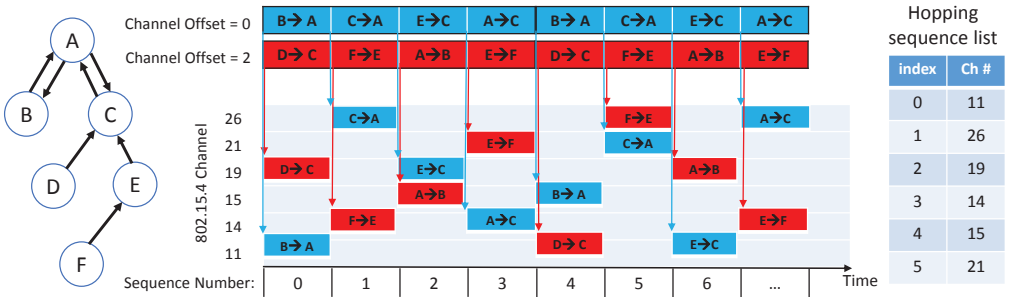


Figure 2.3: An example of the TSCH communications over the timeslot-channel domain.

as on average the failed packet should wait for a longer time to be retransmitted. A successful transmission in a shared timeslot resets *BE* for the transmitting node to a predefined minimum value.

The channel hopping technique of TSCH uses frequency diversity to mitigate blocking of wireless links due to interference and multipath fading. In this technique, at the start of each timeslot each node hops to a new frequency channel based on a predefined *Hopping Sequence List* (HSL). Thus, successive communications of a link are done over different channels (Figure 2.3). This eliminates blocking of wireless links that is caused by repeated dropping of packets due to interference on one operating channel or repetitive destructive multipath fading.

TSCH uses a global timeslot counter in the network that is called the *Absolute Sequence Number* (ASN). By use of the ASN and a global HSL, each node computes the operating channel of each timeslot using Equation (2.1).

$$Channel = HSL[(ASN + Channel\ Offset)\%|HSL|] \tag{2.1}$$

$|HSL|$ is the number of channels in the HSL. Different *Channel Offsets* can be assigned to different timeslots in the network to enable parallel communications in one timeslot on multiple channels (*Channel Offsets* 0 and 2 in Figure 2.3). The HSL may include all or a subset of channels defined by the IEEE 802.15.4 PHY, as determined by the upper layers in the protocol stack.

Assignment of timeslots and channel offsets to the links and extracting a communication schedule, called scheduling, is left to the upper layers in the protocol stack (i.e., a sublayer between network and MAC layers). The communication schedule has a high impact on the performance of the WSN. There are several TSCH scheduling algorithms such as [8, 13, 25, 44, 55, 56] that aim at improving reliability and/or latency. These scheduling algorithms usually assume the network communication topology (that is determined by the network layer) as their input, and their output differs for different input topologies. Figure 2.3 shows an example TSCH communication schedule over time and channels for a given network.

Some of the wireless nodes in a network, known as coordinators, handle the network formation and setup. Coordinators use beacon transmissions for this purpose. TSCH de-

fines the Enhanced Beacon (EB) that is an extension of the IEEE 802.15.4 beacon frame to construct application-specific beacon content. It provides a means for application-specific information provided by higher layer protocols to be included in beacons. This data includes the HSL and network timeslot schedule that is provided by the higher layers. EB transmission can be periodic and/or aperiodic.

In this thesis, we focus on the TSCH protocol from different aspects, to improve its dependability as the MAC layer of an automotive WSN. These aspects include the timeslot structure to use the available bandwidth efficiently, HSL selection to mitigate external interference, and scheduling regarding the network topology.

2.3 Performance Metrics

Specific QoS requirements should be met for a dependable WSN to be used for automotive applications. For each QoS metric, depending on the type of wireless node and the application scenario, different requirements may be defined. Usually, an application defines a set of high-level requirements for a specific scenario. These high-level requirements are independent of the network architecture and target the end-to-end communications between the wireless nodes and the gateway node. End-to-end reliability, latency, throughput, and energy consumption are some of the important QoS metrics. This thesis develops mechanisms on top of the TSCH MAC in order to build a dependable protocol stack for automotive WSNs. To evaluate the detailed performance of these mechanisms with respect to the QoS requirements, we consider some low-level metrics as well. These metrics include link-level reliability, link-level latency, and maximum length of burst packet losses which reflect the behavior of lower layers of the networking protocol stack. This section introduces all the used QoS metrics that are used throughout this thesis, and presents their calculation method.

2.3.1 Communication Success Ratio

Wireless communications may experience packet losses due to link failures caused by collisions, interference, and synchronization problems. Packet losses may lead to data losses, if the communication protocol does not take care of them. Communication success ratio of a network shows the ability of the network to handle link failures. Collision avoidance and interference mitigation at the MAC layer and retransmissions at higher layers are examples of the techniques that can be used to improve the communication success ratio of the network.

Different metrics can be used to measure communication success ratio at different levels. At the link level, Packet Reception Ratio (PRR) and Packet Delivery Ratio (PDR) can be used. PRR is the percentage of the received packets at the destination side of a link over the total number of packets that are transmitted by the source node of that link, in a certain period of time. PRR shows the directional success ratio of a link and ignores the effect of the acknowledgement packets in the communication process. This metric evaluates the link reliability against the failures at the point of the receiver node, caused by interference or collisions. This is useful when the physical layer behavior and low-level MAC mechanisms are studied and packets are important as a communication

unit. PDR is the percentage of successfully acknowledged packets over the total number of transmitted packets. This metric can be used to evaluate the performance of the MAC layer mechanisms that are used to control the communication flow on a link.

At higher levels, Data Delivery Ratio (DDR) can be used to evaluate the reliability of end-to-end communications. This metric shows the percentage of data units that are successfully delivered to their destination in the network (e.g., the sink node), over the total number of data samples that are generated by a source node in a period of time. Since application-to-application communications are considered for this metric, it gives the performance of the whole protocol stack, including MAC and routing protocols.

As a network is running, communication success ratio in terms of PRR, PDR, or DDR, can be calculated using a simple moving average over the status of recent communications (failure of success) during a window $\mathcal{W}_{SMA}$. The moving average window $\mathcal{W}_{SMA}$ determines whether a short-term performance (lower values) or a long-term performance (higher values) is of interest. The provided performance by the network, for each one of the introduced metrics, should be higher than the required QoS by the application to indicate the correct service of the network. However, considering the stochastic behavior of wireless communications, which is mostly caused by variable interference, such a requirement cannot be guaranteed all the time. Accordingly, in some periods the performance of the network may go lower than the required value. These requirement misses should be acceptable by the application (soft real-time applications are considered in this thesis). As an example, an automotive application may require DDR of a connection from a sensor to the central ECU (calculated with a predefined $\mathcal{W}_{SMA} = 100$ over time) to be higher than 80% in at least 90% of 1000 consecutive communications at any point of time. These kind of requirements show the reliability and availability of the system, which indicates the continuity of and readiness for correct service. These requirements are usually different for different sensor types and applications. For performance evaluations of this thesis, we report the performance of different proposed techniques using introduced metrics. Accordingly, different types of requirements can be applied to the results to study dependability of the network for different applications.

For our simulation-based performance evaluations, we also introduce Packet Reception Probability (PRP). PRP is the probability of one successful packet transmission for a given noise power during transmission of each bit of that packet in simulations. PRP is a probabilistic value which is extracted based on the Signal-to-Noise Ratio (SNR).

2.3.2 Maximum Length of Burst Packet Losses

Communication disconnections are one of the important problems that affect the dependability of the network. The length of burst packet losses is the number of consecutive packets losses over a link. This metric shows the maximum time that two nodes cannot communicate over a link in a period of time. This is not shown by the communication success ratio metrics. At a higher level, the length of burst data losses is the number of consecutive data units lost over a network connection. Usually, the length of burst data losses is the visible metric to the application and it should be lower than a value predefined by the application.

2.3.3 Latency

For most of the monitoring applications, the validity of the sampled data changes after a specific amount of time and data may even be of no interest anymore. The latency of a data communication is the time between data sampling at the source node and its reception by the destination node. This includes the time that a packet is on air and the time that it is in the MAC buffer of the forwarding nodes along the communication path, waiting to be transmitted. The latter time is larger and varies more and is affected by the communication congestions, reliability issues, and routing decisions. Different techniques at different layers of the networking protocol stack can be used to reduce this part of the latency. Note that latency can be calculated only for the items delivered to the destination. Lost packets are not included in latency calculations.

Depending on the type of samples or actuation command, latency constraints may differ. This is because of the variation in the nature of the sampled data by different types of sensor nodes. The way that the system treats the latency misses for each type of samples may also differ. Some applications may still use the sampled data, even if the delivery latency is more than the latency constraint. These samples usually have lower usefulness and degrade the system's QoS. For some other applications, missing the latency deadline by a data packet leads to its expiration. These deadline misses reduce the reliability of the communications, as it is treated as a failure. As we consider soft real-time applications in this thesis, we consider the former type of applications in our work. Accordingly, in this thesis, we use the average latency of the end-to-end communications for evaluating our proposed mechanisms. Box-plots are also used to show the distribution of latency for different packets transmitted over a link or different end-to-end connections.

2.3.4 Energy Consumption

Considering small batteries as the power source of most of the wireless nodes in an automotive WSN, the energy consumption of nodes is of high importance to provide a long life-time. There are four main energy consuming parts in a wireless sensor node; processor, memory, sensors, and wireless radio. Considering the low-power nature of the wireless sensors, they are usually equipped with very low-power processors, sensors, and small memories to keep the energy consumption as low as possible. The wireless radio is usually the dominant energy consumer part of the wireless nodes [36]. In this thesis, we ignore the energy consumption by the processing parts of the wireless sensors and focus on the radio energy consumption. This is because the difference in energy consumption of different communication protocols is mostly because of their difference in using the radio transceiver.

The radio energy consumption highly depends on the communication protocols on the MAC and network layers, as they impose packet transmissions and receptions. As an example, transmission collisions in a dense network make retransmissions necessary and lead to a higher power consumption. This can be prevented by using a MAC protocol that prevents the collisions. Thus, employing efficient MAC and routing protocols is necessary. Accordingly, in this thesis we define energy consumption as the energy that is consumed for the communications by the wireless radio. The energy consumption for

a given number of packet communications (E_{comm}) can be extracted by Equation (2.2).

$$E_{comm}[J] = \left(I_{Rx} N_{Rx} T_{listen} + I_{Tx} N_{Tx} T_{Tx}\right) \times V_{cc} \tag{2.2}$$

where I_{Rx} and I_{Tx} stand for the radio transceiver current in receive and transmit modes, respectively. N_{Rx} and N_{Tx} reflect the number of occurrences of each operation in the experiment duration and V_{cc} represents the operating voltage of the transceiver. T_{Tx} represents the duration of a full packet transmission, and T_{listen} is the duration that the receiver should listen to the medium to receive a packet or sense its condition. We ignore the radio sleeping energy consumption in our calculations, as it is negligible compared to the communication energy consumption [50, 53].

2.4 Performance Evaluation Methods

2.4.1 Experiments and Simulations

To examine that the performance of a new mechanism is as expected, it should be implemented and evaluated for different system configurations and conditions. Real-world evaluations are usually done using simplified systems with lower capabilities, so that the measured performance only shows the effect of the applied mechanism on the system performance. Furthermore, implementing a realistic system with the full functionality is usually not feasible for research purposes. In some cases, it is even not possible to perform multiple experiments under the same system condition to compare the results of different evaluations for different configurations. This is because of the environment effect on the behavior of the system, that is not predictable. In this case, computer simulations are used to simulate the behavior of the system and environment to evaluate the performance of the system. Although real-world experiments provide more realistic results, simulations make the system evaluation much easier and faster than the real-world experiments. Moreover, there is more control on the system's configurations and setup. In some cases, simulations can even extract some performance metrics that can hardly be measured in the real-world systems. On the other hand, simulations cannot precisely imitate behavior of the real-world systems and environment.

We use both real-world experiments and computer simulations to evaluate the performance of our proposed mechanisms for automotive WSNs. Experiments prove the feasibility of the proposed mechanisms, and simulations extract their performance in more detail. Table 2.1 provides a short overview of advantages and limitations of all the used performance evaluation methods in this thesis. We use three types of experiments in this thesis. The first and second type of experiments deploy wireless nodes in controlled environments with no or very low interference, in particular in, an anechoic chamber and an empty office after working hours. These setups are used to evaluate the proposed mechanisms for which their performances are not affected by the structure of a vehicle. We use the anechoic chamber setup for an evaluation that requires no reflection and diffraction in the environment. This setup is more difficult and time consuming than the office setup. The third type of experiments are performed by deploying wireless nodes inside a car. This setup is used to evaluate a mechanism that is proposed to extract the proper topology for an automotive WSN. This mechanism uses

Table 2.1: Advantages and limitations of the used performance evaluation methods.

Performance evaluation method	Advantages	Limitations/disadvantages
Office experiments	- Easy to repeat - Easy to set up	- Different environment than in-vehicle - Little unwanted interference
Anechoic chamber experiments	- No unwanted interference - No reflection and diffraction for special experiments	- Difficult and time consuming to set up - Different environment than in-vehicle
In-car experiments	- Effect of the car body and positioning of nodes can be studied	- Limited flexibility - Difficult to set up - Not repeatable with the same interference
Matlab simulations	- Use of real-world interference data set - Repeatable with the same interference - Very fast	- Single-link simulations - High-level simulation - Different environment than in-vehicle
COOJA simulations	- Full protocol simulation - Placement of nodes is configurable - Very easy to repeat - Many performance details are available - Flexible	- Real-world interference data set cannot be used - Different environment than in-vehicle

the quality of network links, which are affected by the positioning of nodes inside the vehicle, to decide on the network connections. We also use two types of simulations in this thesis. For the first type of simulations, we implemented a simulation framework in Matlab to model the communication behavior of a single wireless link according to the communication timings of the TSCH protocol. We use the measured real-world in-vehicle interference data set in Chapter 3 as the input of this simulation framework. This enables us to study the performance of TSCH protocol under real-world in-vehicle interference in a controlled way. The second type of simulations uses the COOJA [54] network simulator. This simulator enables full protocol simulation and very flexible configuration of the network. It helps us to study the performance of the network for different configurations and in more detail.

Considering a vehicle on an urban road, it is in the communication range of some interference sources (e.g., Wi-Fi networks) at all times. Thus, every few seconds a moving vehicle goes into the range of some new interference sources that interfere with one or multiple different IEEE 802.15.4 channels. For example, assuming that the wireless signal of a Wi-Fi device is visible over a range of up to 50 meters, and the car moves with a speed of 36 km per hour, the Wi-Fi interference would be visible to the automotive WSN for 5 seconds. Accordingly, we deploy a number of noise generators in our experimental setups to mimic the interference of the automotive environment. The number of noise generators and their behavior can be configured to model different driving scenarios. Using these noise generators enables us to perform repeatable experiments for different techniques and network configurations.

2.4.2 Experimental Platforms

To implement our techniques and evaluate their performance, we use two different types of IEEE 802.15.4-based wireless nodes, namely Atmel ATMEGA256RFR2 Xplained Pro kit [50] and NXP JN5168 dongles [53]. The Atmel kit includes an ATmega256RFR2 chip which integrates an 8-bit AVR microcontroller with 256 kB in-system flash memory and 32 kB internal SRAM. It also embeds a 2.4 GHz RF IEEE 802.15.4 compliant transceiver with -100 dBm sensitivity and maximum output power of 3.5 dBm. The RX current of this radio is 12.5 mA and the TX current is 14.5 mA for the TX output power of 3.5 dBm. The supply current of this radio at the sleep mode is 0.02 μA which is negligible compared to other modes.

For the Atmel boards, we implemented the TSCH slotted communications and channel hopping on top of the Atmel implementation of the basic IEEE 802.15.4 MAC. Our implementation follows the default TSCH timings, defined in the standard. This platform was developed since there was no official implementation of the TSCH MAC by the beginning of 2015. It is used for evaluation of the channel whitelisting technique that is presented in Chapter 4.

The NXP JN5168 dongle [53] includes a wireless microcontroller which integrates a 32-bit RISC processor with 256 kB embedded Flash and 32 kB RAM. It also includes an embedded 2.4GHz IEEE 802.15.4 compliant transceiver with -95 dBm sensitivity and RX current of 17 mA and TX current of 15 mA with a maximum transmit power of 2.5 dBm. This platform is supported by the Contiki [24] Operating System (OS). We use this platform for the evaluation of other techniques that are proposed in this thesis.

Contiki is an open source OS for the low-power and memory-constrained WSNs. Contiki provides multitasking and different components to develop wireless networking mechanisms. On a higher level, it also provides different networking mechanisms such as TCP/IP, fully standard IPv6 and IPv4. This platform is designed to run on different types of hardware devices with low capabilities. TSCH was added to the official Contiki version 3.1 on January 2017 and is fully based on the standard definitions.

We use JN5168 dongles, that run Contiki OS with TSCH protocol at the MAC layer, for a set of our real-world experiments. This gives us the opportunity to compare our work with other related works for which the Contiki implementations are already available. According to the requirements to examine the feasibility of each proposed mechanism, we use different numbers of wireless nodes for the evaluation of each one.

2.4.3 Simulation Environment

To study the performance of the proposed mechanisms in more detail and with more configurations, we use simulations next to the real-world experiments. One of the advantages of using Contiki as the OS for the wireless nodes is that it enables simulation of the same embedded code that is prepared for the hardware platforms on its computer simulator, COOJA [54]. COOJA is a Java-based simulator that emulates the behavior of the real-world wireless nodes to simulate the behavior of a network. Using this simulator, a user can define different types of wireless nodes with different applications in a network, and place them at the desired position. While COOJA provides a nice visualization of the network and communications, it also enables configuring the network parameters (e.g., transmission range) at runtime.

There is stochastic behavior in the wireless communication protocols. Accordingly, results of a simulation setup may deviate in different running iterations. Moreover, the simulation results may deviate from the real-world average performance of the network. To make the simulation results statistically more reliable, we repeat the simulations for each scenario multiple times with different seed value for the random number generator, so that different patterns of random values are generated. Accordingly, the presented simulation results in this thesis are the average results of multiple simulations that are done for the same setup.

3

Interference in In-Vehicle Networks

3.1 Overview

The IEEE 802.15.4 standard operates in the unlicensed 2.4 GHz ISM band. This band is also used by other standards including IEEE 802.11 Wi-Fi [5] and IEEE 802.15.1 Bluetooth [1]. Figure 3.1 shows the allocated frequencies to these three protocol standards in the ISM band. Considering the ever increasing number of Wi-Fi and Bluetooth-enabled devices, this common usage of the frequency band leads to cross-technology interference and packet losses. Moreover, there are a number of other devices such as Bluetooth Low Energy (BLE) [16] devices, cordless phones, and microwave ovens that operate in this frequency band and cause wireless interference for the mentioned technologies.

Considering the low transmission power used in WSNs, the IEEE 802.15.4 networks are expected to be affected considerably by the other coexisting technologies. There are several experimental and analytical studies on the coexistence of the IEEE 802.15.4 standard and other technologies. However, none of them considers the automotive conditions and its effect on the quality of the links in WSNs. Moreover, most of the available studies focused on single-channel communication protocols and only few of them consider multi-channel communication protocols such as IEEE 802.15.4 TSCH.

In this chapter, we investigate the 2.4 GHz wireless interference behavior in vehicles using real-world experimental measurements. Our goal is to understand how is the interference dynamism over time and its distribution over the IEEE 802.15.4 channels in different driving environments. This information can be used to develop proper techniques to mitigate the effect of this interference. We categorize the cross-technology interference in in-vehicle environments into interference of in-car and out-of-car sources. Accordingly, we consider different vehicular scenarios and measure the interference power on all the 16 channels of IEEE 802.15.4 in the 2.4 GHz band. The measurement results are used in a simulation framework to analyze TSCH behavior under different interference scenarios.

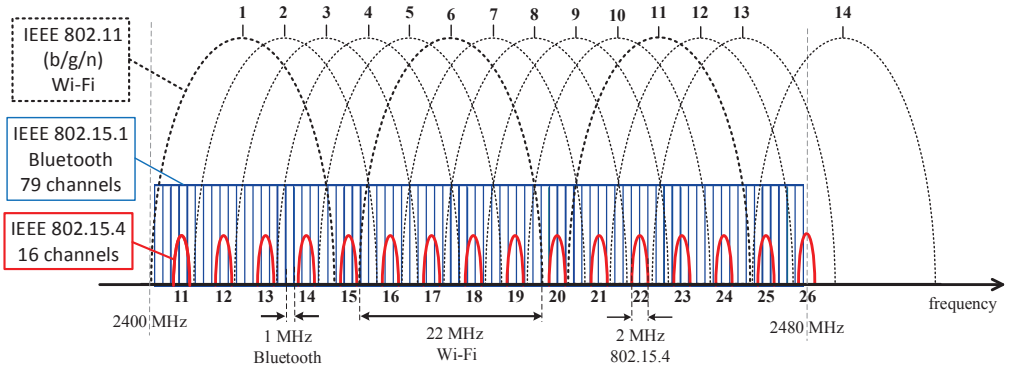


Figure 3.1: Usage of 2.4 GHz ISM band.

This chapter is based on publication [73] and is organized as follows. The next section gives an overview of related work about wireless coexistence in WSNs. Section 3.3 presents our measurement setup and scenarios in detail. The measurement results of in-car and out-of-car interference are discussed in Sections 3.4 and 3.5, respectively. The performance of TSCH communications under measured in-vehicle interference is studied in Section 3.6. Section 3.7 summarizes the findings in this chapter.

3.2 A Review of Wireless Interference in WSNs

The ever increasing number of the 2.4 GHz ISM band users makes wireless interference of coexisting wireless devices a challenge, especially for low-power IEEE 802.15.4 WSNs. The IEEE 802.15.4 standard document [2] provides estimation of packet error rate of this standard under IEEE 802.11b [5], IEEE 802.15.1 [1], and IEEE 802.15.3 [6] networks using coexistence simulations. Some work has been done on the coexistence of IEEE 802.15.4 with other standard wireless protocols using experiments and analytic modeling. Experimental studies presented in [10, 37, 41, 60] mainly measure and report the impact of coexistence on the network performance metrics such as PRR and latency. The authors of [52] and [67] provide analytic models of the coexistence of IEEE 802.15.4 under IEEE 802.11 interference, based on the transmission patterns of both technologies. A radio link quality estimation survey in IEEE 802.15.4 WSNs is provided in [12]. The authors present the observation that the external interference of Wi-Fi and Bluetooth has a strong impact on the quality of IEEE 802.15.4 links, but the communications of Wi-Fi and Bluetooth are less affected by an 802.15.4 network. The authors of [51] use mathematical analysis as well as real-world experiments to study the coexistence between IEEE 802.15.4, BLE [16], and IEEE 802.11b [5]. They show that BLE is affected more by IEEE 802.15.4 interference than vice versa. However, they confirm that the effect of Wi-Fi interference on the quality of IEEE 802.15.4 links is more than its effect on the BLE links.

Different WSN operating environments may lead to different coexistence and interference conditions. While some studies such as [10, 52, 59, 90] focus on the general

coexistence, others consider specific environments such as buildings [37, 86], industrial [15], outdoor [38], and body [41] environments. There are also studies on the wireless coexistence in automotive WSNs. The authors of [91] consider Wi-Fi and Bluetooth as the most likely interferers for the IEEE 802.15.4-based automotive WSNs. They provide measurements and analysis for interference of these technologies on a single channel of IEEE 802.15.4, done in an RF anechoic chamber. This makes this work similar to general coexistence studies, skipping the real-world conditions. The authors of [20] do some measurements for a static in-vehicle scenario. They place some IEEE 802.15.4 sensor nodes in different parts of a car, and investigate the performance of different single channel links between them under Bluetooth communications. The results are expressed in terms of Packet Error Rate (PER) and average/peak latency. These studies only addressed the coexistence effect of devices inside a vehicle on single channel automotive WSNs. However, automotive WSNs (that are operating in single or multiple channels) may also experience interference from devices out of the vehicle.

A channel quality measurement data set for industrial wireless environments is presented in [15]. These kind of public data sets are useful for interference modeling and network performance simulation based on real-world situations. However, the authors of [15] note that these data sets are limited to the office, laboratory, and industrial environments and there is nothing like this for in-vehicle environments.

In this chapter, we focus on the multi-channel automotive WSNs and the effect of cross-technology interference on them. We consider different real-world measurement scenarios and drive a car in different places with various interference conditions. For each scenario, we perform a set of interference measurements on all the IEEE 802.15.4 channels in the 2.4 GHz band, and provide a data set for in-vehicle environments. Such data set can be used to estimate the performance of automotive WSNs. We also evaluate the performance of a TSCH link under real-world interference using simulations and the extracted data set.

3.3 Measurement Setup and Scenarios

We use real-world measurements to capture the wireless conditions of all the IEEE 802.15.4 channels in in-vehicle environments. In this section, we describe the employed measurement setup and its requirements. We also present the experimental scenarios that are used to capture the interference behavior.

3.3.1 Measurement Requirements and Setup

To perform noise measurement on the IEEE 802.15.4 channels, we need to sample each channel continuously. Each channel experiences dynamic energy levels for different durations of time. This is caused by packet transmissions of different coexisting technologies. Considering the broad usage of Wi-Fi and Bluetooth-enabled devices, Wi-Fi and Bluetooth are considered to be two coexisting technologies that have the most impact on the IEEE 802.15.4 automotive WSNs. The data rate, packet size, and bandwidth usage of these standards vary from each other and even from version to version and application to application. Therefore, the sampling method, rate, and duration can have direct

impact on the extracted behavior of the wireless channels. Considering these facts, we need to sample the medium with the highest possible rate. Each sample should reflect the medium quality during the sampling duration.

To measure the interference, we used Atmel Xplained Pro kits. We assign one kit to each one of the 16 IEEE 802.15.4 channels on the 2.4 GHz ISM band to measure the noise level of that channel. All the AVR kits are placed next to each other in a passenger car. The measurement results are continuously streamed to a laptop via different UART serial connections. We use hardware ED, defined in the IEEE 802.15.4 protocol, to measure the quality of the wireless channels. In the Atmel chip, each ED has a value in the range of [-90, -10] dBm.

Since TSCH may use multiple channels for communications at the same time, knowing interference condition of all channels at the same point of time is necessary to analyze the behavior of communications. Thus, sampling of different channels should be synchronized to correctly show the interference behavior. Clock drifts of different AVR chips make a one-time initial synchronization useless. Accordingly, we use wired signaling between kits to synchronize them at the beginning of each sampling interval (one ED sampling per interval). One of the kits works as master and triggers an output pin at the start of each sampling interval. Other nodes get this signal as input and start each sampling period when it is triggered. We set the sampling period to 500 μs which is enough to do an ED (128 μs) and send the result to the computer via UART. On the computer side, we use Matlab to collect the sampling data that is sent by individual kits via UART connections.

3.3.2 Measurement Scenarios

We categorize the interference sources for automotive WSNs into in-car and out-of-car sources. For each category, we perform several measurements using different real-world scenarios. For in-car interference sources, each scenario is designed to investigate the effect of one common source of interference and/or application. In this case, we picked three measurement scenarios to study the behavior of Wi-Fi and Bluetooth transmissions. The three scenarios are:

1. Bluetooth connection of a mobile phone and the audio system of the car with an audio streaming application,

2. Bluetooth connection between two smart phones with a file transfer application which requires more bandwidth and handshaking than that of first scenario,

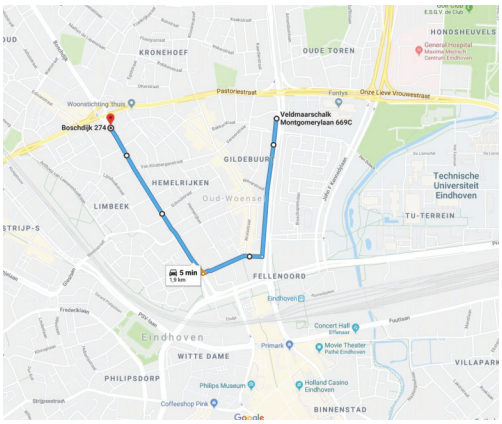
3. Wi-Fi connection between two smart phones with a file transfer application.

The out-of-car interference is caused by the devices that are operating out of the car along the roads or in other cars. We defined four scenarios in this case that are:

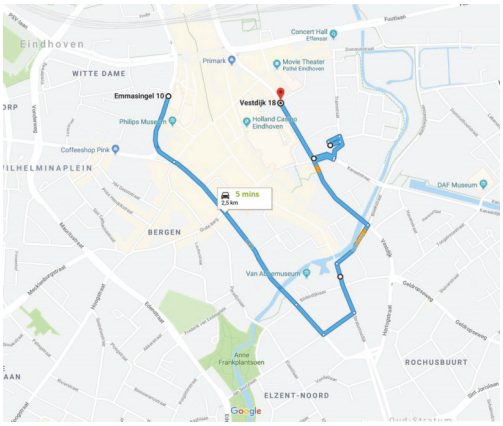
1. Driving along a route near some apartments (Figure 3.2(a)),

2. Driving along an office area downtown (Figure 3.2(b)),

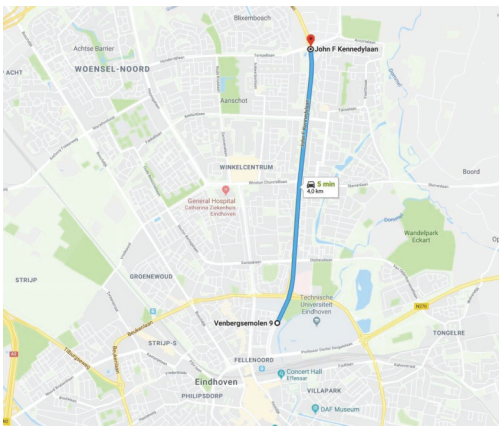
3. Driving in a suburb area (Figure 3.2(c)),



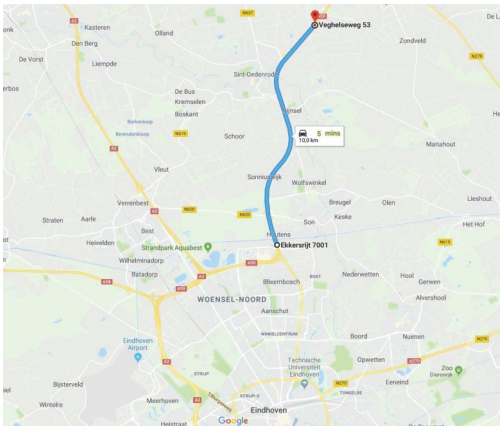
(a) Driving near apartments, 1.9 km, 5 minutes



(b) Driving downtown, 2.5 km, 5 minutes



(c) Driving in suburb, 4 km, 5 minutes



(d) Driving on highway, 10 km, 5 minutes

Figure 3.2: The driving scenarios that are used for out-of-car interference experiments.

4. Driving along a highway with no buildings around (Figure 3.2(d)).

In the following sections, we present the measurement results of different scenarios in each interference category.

3.4 In-Car Interference Measurement

To study the interference behavior of in-car sources, we parked the car in an open space area with no construction within 0.5 kilometer. Using sniffers, it was confirmed that the selected environment has negligible external interference on the 2.4 GHz ISM band. We performed measurements for the three in-car scenarios with Wi-Fi and Bluetooth devices in the car. Each measurement is performed for 5 minutes which leads to 600k samples per channel. In the following, we discuss the result of each measurement in detail.

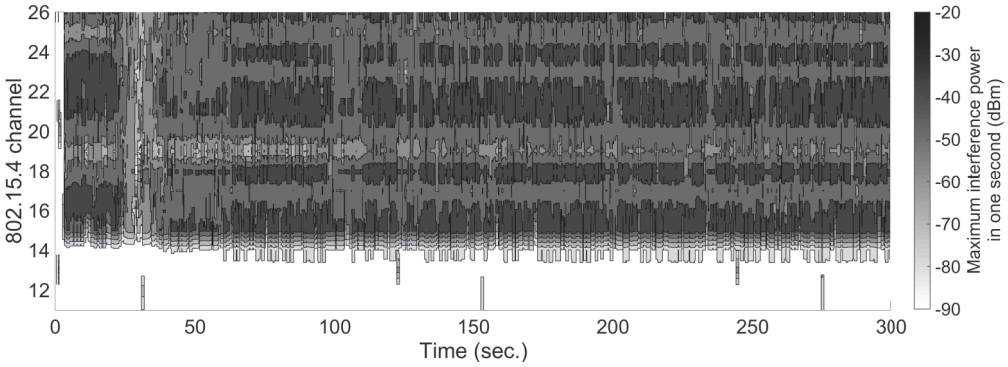


Figure 3.3: Interference behavior of Bluetooth audio streaming on the IEEE 802.15.4 channels over 300 s using contour plot.

3.4.1 Bluetooth Audio Streaming

In this scenario, we used a mobile phone to stream audio to the audio system of the car using Bluetooth version 4.0. We placed the phone on the dashboard of the car, with 2.5 meters distance from the interference measuring motes. Figure 3.3 uses a contour plot to show the distribution of interference power over time and different channels. Each color in the plot reflects the maximum power of the captured Bluetooth interference on a channel during a period of one second. In contour plots, the width of the samples' color on the horizontal line shows the repetition of samples with that power level in that channel over time.

The first observation of Figure 3.3 is that there is no Bluetooth interference on some of the IEEE 802.15.4 channels (first four channels in this experiment). This is because of the blacklisting method that is used by the Bluetooth channel hopping module. It should be considered that this blacklisting method may be different in different Bluetooth devices, from version to version, and vendor to vendor. The Bluetooth channel hopping module can also be pre-programmed to not use some parts of the frequency band to prevent cross-technology interference with in-range devices.

The second observation of Figure 3.3 is that the usage of different parts of the frequency band is not uniform. For instance, some of the channels, such as channel 22, experience Bluetooth interference with a higher power (darker parts of the plot), while some others, such as channel 19, experience lower power Bluetooth interference. To make it clearer, Figure 3.4 shows the measured noise on channels 19 and 22 during one second. In this example, the measured interference power of Bluetooth transmissions on channel 22 is considerably higher than on channel 19. A possible reason is the cross channel interference and distance between center frequency of the Bluetooth operating channels and the measured IEEE 802.15.4 channel. Another reason is the multipath fading that affects different frequencies differently. Thus, different adjacent Bluetooth channels can cause interference with different signal powers on an IEEE 802.15.4 channel.

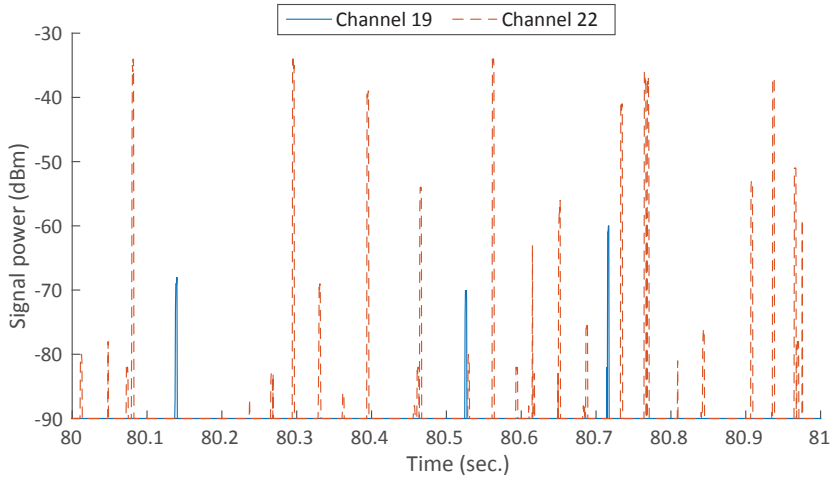


Figure 3.4: Measured Bluetooth audio streaming interference power on channels 19 and 22 over 1s.

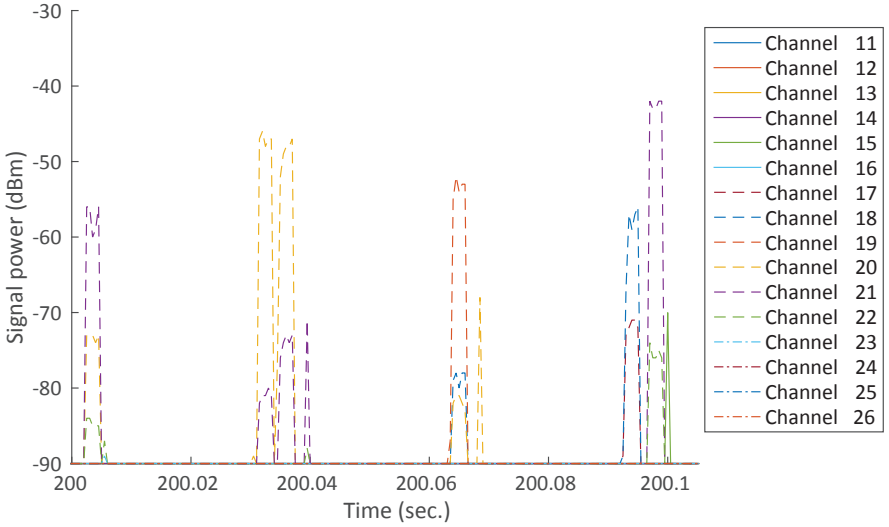


Figure 3.5: Measured Bluetooth audio streaming interference power on the IEEE 802.15.4 channels over 100 ms

Figure 3.4 also shows that the number of interfering samples of the Bluetooth transmissions on the two IEEE 802.15.4 channels are different. In this example, channel 22 experiences interference of 20 Bluetooth transmissions, while channel 19 only experiences interference of 3 Bluetooth transmissions for the same period of time. This shows that the distribution of the Bluetooth interference is not uniform over different channels, and some channels may be occupied more than others. This behavior is caused by the channel hopping of Bluetooth.

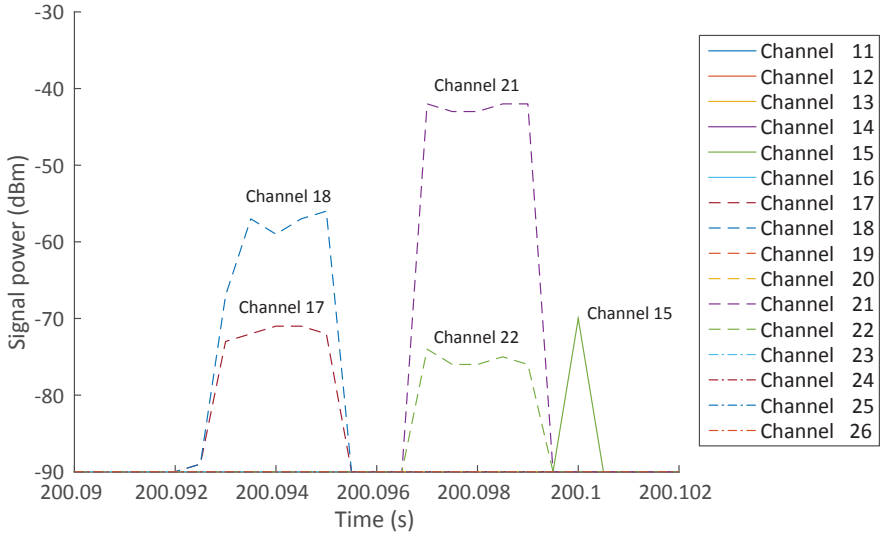


Figure 3.6: Interference of one Bluetooth packet transmission, one retransmission, and one acknowledgement on IEEE 802.15.4 channels.

We have a more detailed look at the Bluetooth interference behavior considering the application of audio streaming that is used in this scenario. Figure 3.5 depicts the Bluetooth interference measured on the IEEE 802.15.4 channels in a 100 ms time period in this scenario. The measured interference follows a periodic behavior with intervals of around 30 ms. Each transmission lasts for 3 ms which is the transmission time of a Bluetooth packet with the maximum size. This shows that the audio streaming application sends periodic packets that require a bandwidth of around 10% of the available Bluetooth bandwidth.

Figure 3.6 shows the measured interference of one complete Bluetooth packet transmission. In this case, we can say that the first Bluetooth packet transmission fails because it is not followed by the receiver's acknowledgement. Thus, the transmitter sends the packet again within a short interval, and in this try, it is followed by an acknowledge packet. As mentioned before, the difference between measured signal powers on different IEEE 802.15.4 channels for a single (or multiple) Bluetooth transmission(s) can be because of the different distance between center frequency of the Bluetooth operating channels and the measured IEEE 802.15.4 channels.

As a conclusion, voice streaming over Bluetooth produces periodic transmissions that lead to non-uniform interference for IEEE 802.15.4 channels. Thus, some of the IEEE 802.15.4 channels may experience less interference than other channels over time. The power level of this interference on different channels is also non-uniform, but on each channel it is often stable over substantial periods of time.

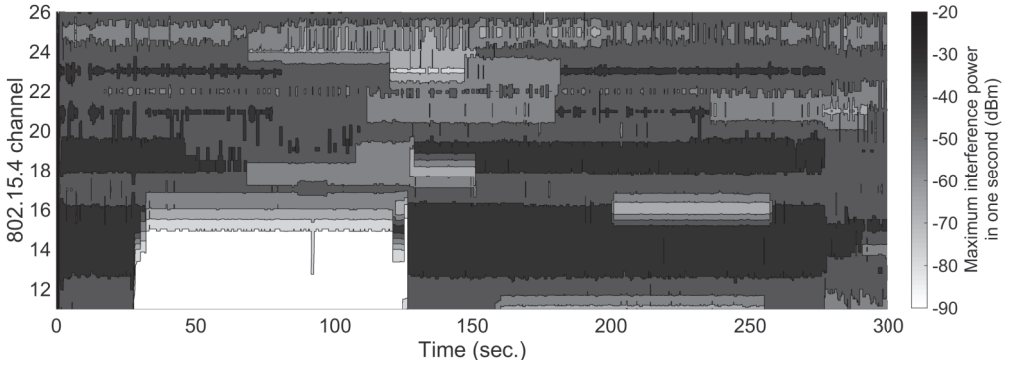


Figure 3.7: Interference behavior of Bluetooth file transfer on IEEE 802.15.4 channels over 300 s using contour plot.

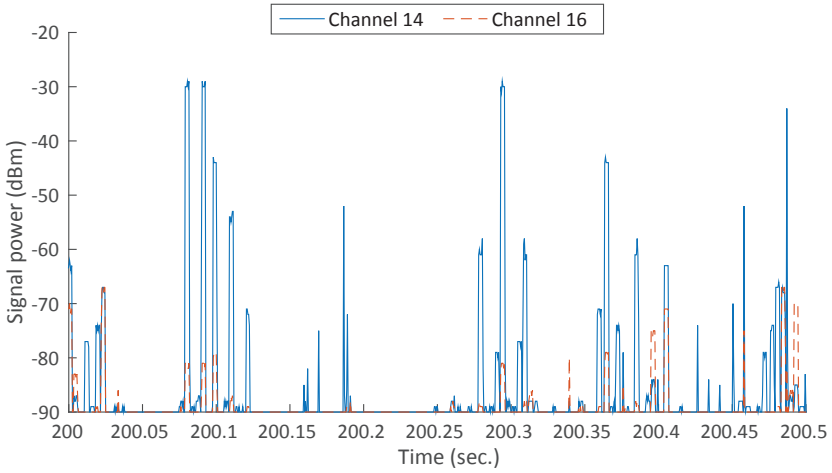


Figure 3.8: Measured Bluetooth file transfer interference power on channels 14 and 16 over 0.5 s.

3.4.2 Bluetooth File Transfer

To study the behavior of Bluetooth interference when it is under high loads, we use two mobile phones to transfer a large file using Bluetooth. The transmitter phone was placed on the back seat of the car near the interference measuring motes. The receiver phone was placed on the dashboard with 2 meters distance from the transmitter phone. As Figure 3.7 shows, as for the audio streaming scenario, some of the channels experience more interference than others. Furthermore, the power level of the interferer signal on a single channel varies over time (compare interference power on channel 14 at $t = 100$ and $t = 200$).

Figure 3.8 shows the measured Bluetooth interference on channels 14 and 25 at $t = 200$ for half a second. This figure shows that the repetition of interferer signals and their power in channel 14 is considerably higher than in channel 16. This is while Bluetooth uses its full bandwidth to transfer data in this scenario (see constant trans-

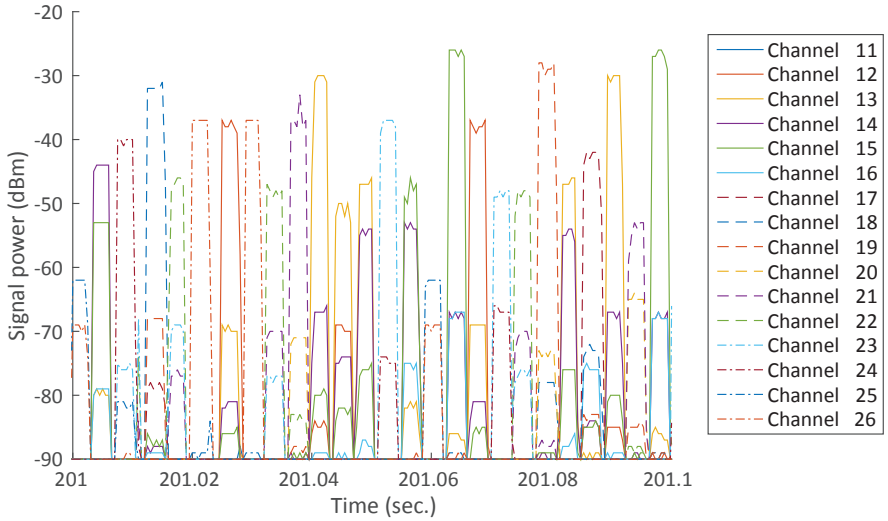


Figure 3.9: Measured Bluetooth file transfer interference power on the IEEE 802.15.4 channels over 100 ms.

missions in Figure 3.9). Considering the results of the first scenario, we can conclude that Bluetooth causes a non-uniform interference over IEEE 802.15.4 channels for different applications with different data transfer rates. The important point here is that the power of Bluetooth interference on each channel is almost stable over substantial periods of time.

3.4.3 Wi-Fi Connection

To study the interference behavior of Wi-Fi devices within the vehicle, we connected two smart phones using a *Wi-Fi direct* connection and use this connection to transfer some large files. One of the phones is used as the transmitter and the other one as the receiver of files. During the experiment, these two phones are placed in different positions inside the cabin. We logged the generated interference of this Wi-Fi connection on the IEEE 802.15.4 channels for 300 seconds.

Figure 3.10 shows the interference behavior over time and channels, using a contour plot. It shows that the Wi-Fi interference mostly affects a few adjacent IEEE 802.15.4 channels and the power of this interference decreases by going far from the center frequency of the Wi-Fi operating channel. This plot also shows some transmissions at other frequencies than the frequency channel that is used for the mentioned Wi-Fi connection. These are probe requests (to perform active scans) and beacons (to advertise a P2P Group) that are done on so-called social channels, namely channels 1, 6 or 11 in the 2.4 Ghz band, by *Wi-Fi direct* devices [87].

In this experiment, the center frequency of the Wi-Fi operating channel is between channels 12 and 13 of IEEE 802.15.4. As Figure 3.10 shows, the interference strength changes over time. These changes are due to the movement of two phones which changes the distance between interferer and sensor nodes. Compared to Bluetooth,

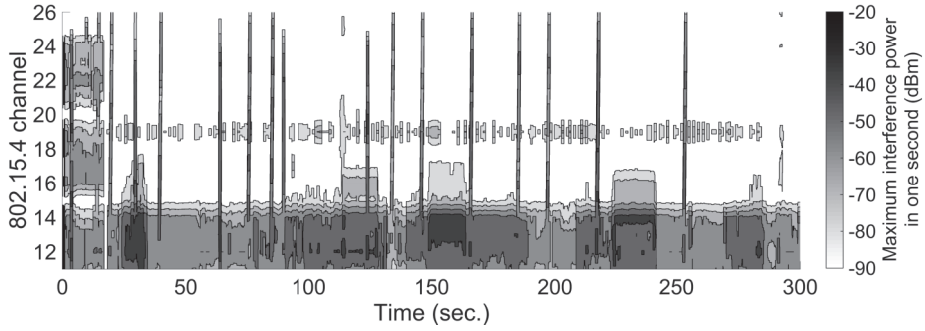


Figure 3.10: Interference behavior over 300 s using contour plot.

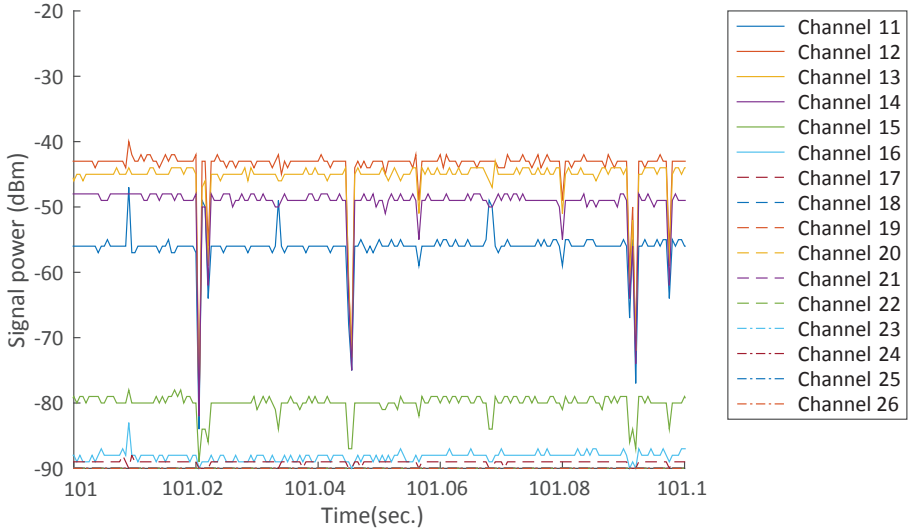


Figure 3.11: Wi-Fi interference power on the IEEE 802.15.4 channels over 100 ms.

the observed interference on each channel is more stable over time. This is because Wi-Fi devices do not use channel hopping and a connection normally uses a fixed channel for communications.

Figure 3.11 depicts the interference of Wi-Fi transmissions on the IEEE 802.15.4 channels over 100 ms. According to this plot, channels 11 to 14 are within the main 22 MHz bandwidth of the Wi-Fi operating channel, while channels 15 and 16 are on the sidebands of the Wi-Fi operating channel. Because the file transfer application uses the full bandwidth of the Wi-Fi connection, the captured interference on each channel is almost constant during the transmission period of a file.

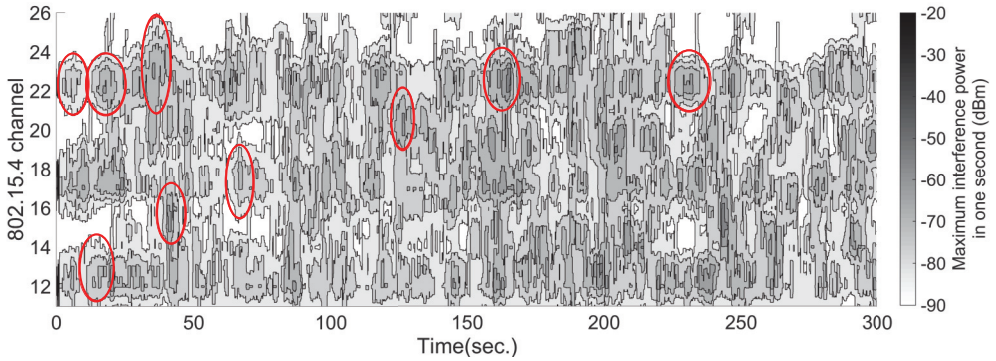


Figure 3.12: Behavior of out-of-car wireless interference near apartments.

3.4.4 In-car Interference Conclusion

In this section, we investigated the interference behavior of two main sources of in-car interference for automotive WSNs. Real-world measurements showed that the power distribution of this type of interference on different IEEE 802.15.4 channels is not uniform. However, depending on the interferer protocol and the used application, the interference power on each channel is almost stable over substantial periods of time (in order of a few seconds). We may conclude that the non-uniform interference over different channels suggests the need for a proper channel whitelisting (or blacklisting) mechanism. These mechanisms should also cope with the dynamism in the quality of each channel over time.

3.5 Out-of-Car Interference Measurement

To study the interference behavior of out-of-car sources, we drove a car in different environments. During these measurements, all the in-car interferers were turned off and the car was driven according to the speed limit in that district. Four scenarios are considered that include apartment area, downtown, suburb, and highway. Considering the higher transmission power of Wi-Fi compared to Bluetooth devices, we expect the Wi-Fi devices on the sides of the roads to be the main source of out-of-car interference. By using a Wi-Fi analyzer application on a mobile phone, we found that the density of Wi-Fi devices in these four scenarios decreases from apartment areas to downtown, suburb, and highways. We drove for 5 minutes in each environment while the interference measuring motes measure the noise power on all 16 channels. Figures 3.12 to 3.15 show the captured interference in different environments using contour plots.

Figure 3.12 shows the interference behavior while driving near apartments with a speed in the range of 10 to 30 kmph. As it was expected, the interference power (maximum -40 dBm) and density in apartment areas is more than other environments. In this figure, there are lots of overlapping ovals with a high power at their centers (some of which are marked by red ellipses). This is because when the car is in the range of one Wi-Fi device and moves toward it, the interference power will be increased and

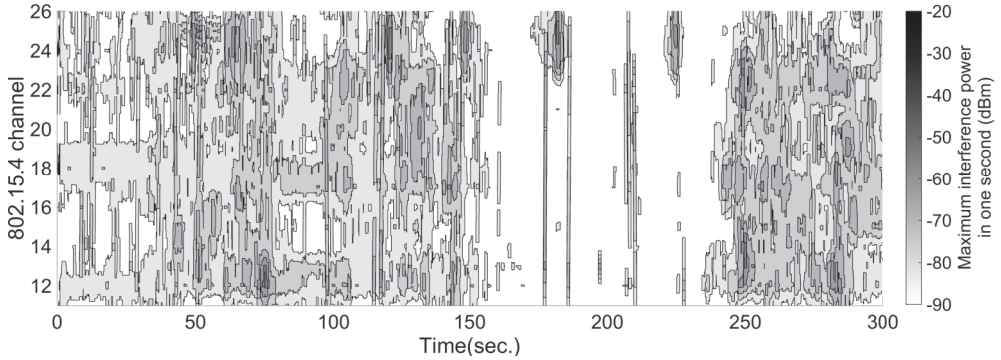


Figure 3.13: Behavior of out-of-car wireless interference in a downtown environment.

vise versa. Thus, the interference of one Wi-Fi device is only visible for a few seconds. This figure shows that at any point of time, the car may be in the interference range of multiple Wi-Fi devices. These Wi-Fi devices may even overlap in the operating channels and each one can affect 2 to 3 IEEE 802.15.4 channels. On the other hand, some of the IEEE 802.15.4 channels are noise free over some periods of time; this can be seen as white spaces on the contour plots.

The downtown scenario (Figure 3.13) has two specific properties. First, the speed of the car is determined by the road traffic and the traffic lights (in the range of 0 to 50 km/h). This affects the time that a car will be in the range of a stationary interferer and thus affects the dynamism of the interference. For example, around time 0s to 30s in the Figure 3.13, the car has been waiting for a traffic light and the interference on channels 12 and 18 is almost stable. This is while from time 250s to 300s, the car has been moving along the street and the observed interference is relatively more dynamic. The second property is that the car moves next to other cars in the street in the same or opposite direction. These neighbor cars may carry some devices that are operating in the 2.4 GHz ISM band (Bluetooth, Wi-Fi, etc.). This may lead to long or short-term interference. The vertical bars in Figure 3.13 may be because of such interferences. These bars can be due to Bluetooth transmissions in the neighbor cars which affect most parts of the frequency band, because of the fast channel hopping of the Bluetooth protocol. Due to the low communication range of Bluetooth devices, this interference is only visible for a short time period when cars are in a distance of few meters.

In the third scenario, the test car is moving in a street in a suburb area with an average speed of 50 km/h. As it is clear in Figure 3.14, the interference power in this area (maximum -60 dBm) is less than apartment and downtown areas and there is more noise-free area left in the channel-time space. This is because of lower density of houses in suburb areas which leads to lower density of interferers. This also causes longer distances between stationary interferers and the car, which reduces the power of the observed interferer signal.

Figure 3.15 shows the observed interference on a highway while the car is moving with a fixed speed of 120 km/h. In this scenario, the stationary interferers play the least role (only near the gas stations). The main source of interference in this scenario is

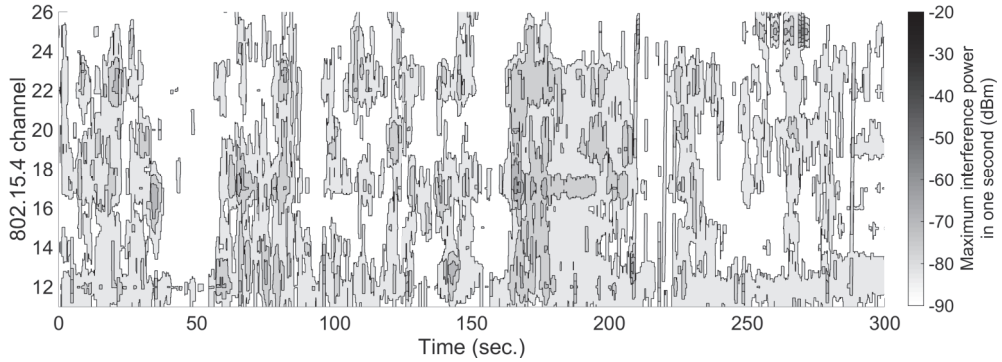


Figure 3.14: Behavior of out-of-car wireless interference in a suburb environment.

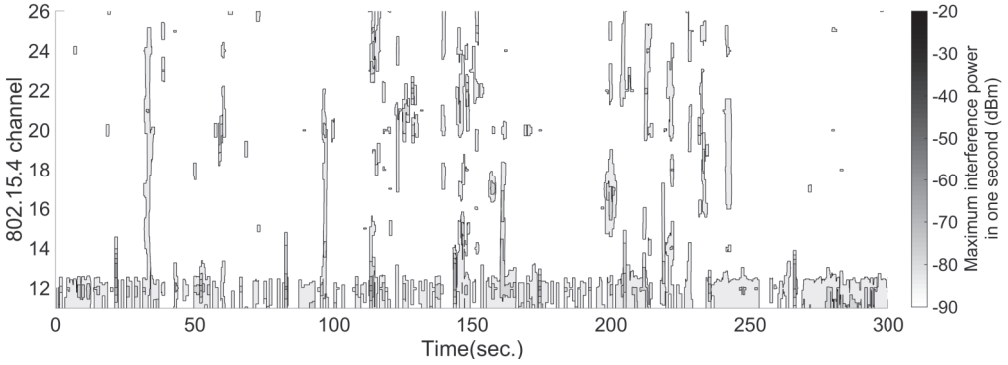


Figure 3.15: Behavior of out-of-car wireless interference in a highway

the interferer devices in neighbor cars. Considering the short time and high distance adjacency of cars in a highway, these devices cause low power (maximum -70 dBm) and short term interference to the automotive WSN. However, some IEEE 802.15.4 channels such as channels 11 and 12 may experience more interference than other channels, as they are conflicting with the Wi-Fi channel 1 that is usually the default Wi-Fi channel of the automotive on-board computers.

Considering the mentioned observations of the out-of-car interference behavior, it can be concluded that in-vehicle wireless sensor communications may face serious problems in city environments if the operating channels and transmission power are selected blindly. In the next section, we study the effect of such interferences on the performance of the TSCH protocol by using probabilistic communication models and the collected interference data set. This data set and the simulation scripts are publicly available online [62].

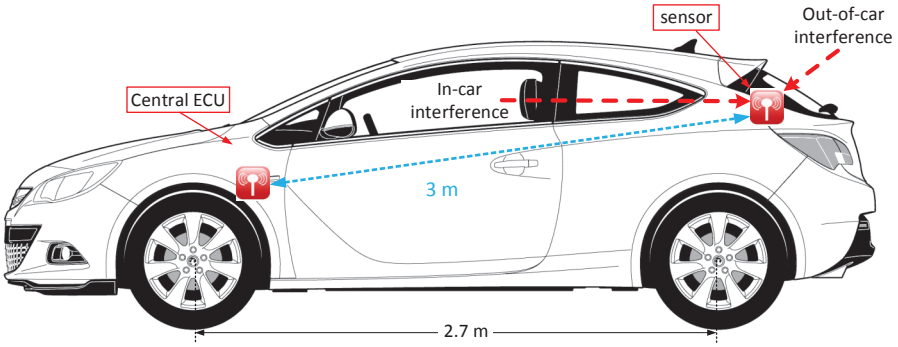


Figure 3.16: Interference model for packet reception probability computation.

3.6 Automotive WSNs Simulation Model

In this section, we develop a simulation framework that uses the measured interference data set to evaluate the performance of the IEEE 802.15.4 TSCH communications. We use a simple model to extract the communication behavior of a single wireless link in a car (shown in Figure 3.16). In this model, there is a wireless sensor node placed exactly where we placed the interference measuring motes, and is connected through a wireless link to an ECU inside the dashboard. Considering a direct wireless link from the ECU to the sensor node, the received signal power (P_{rx}) at the sensor node can be computed as:

$$P_{rx}[dBm] = P_{tx}[dBm] - PL(d)[dB] \tag{3.1}$$

where P_{tx} is the power of the signal at the transmitter (central ECU) and $PL(d)$ is the path loss at distance d. We use the path loss model (Equation (3.2)) introduced in [2] for short range communications at 2.4 GHz band.

$$PL(d)[dB] = \alpha\ [20.1 + 10\log(d)] \qquad d \leq 8m \tag{3.2}$$

where α is the path-loss exponent, which has a value equal to 2 for free space and different values for other environments. The typical α for intra-vehicular environments is reported to be around 3.5 [14, 21, 65].

The receiver node in our framework experiences interference from sources inside and outside the car. The probability of successful communication at time t is related to the SNR [45] at that time. Here we focus on the effect of interference from coexisting devices. Thus, the *SNR* at time t is given in decibel as:

$$SNR^t[dB] = P^t_{rx}[dBm] - P^t_{intf}[dBm] \tag{3.3}$$

where P^t_{intf} is the interference power at time t at the receiver point within the same bandwidth as P_{rx}. Considering that the distance between the transmitter and receiver in our model is 3 meters, the *SNR* for the given link can be presented as:

$$SNR^t[dB] = P^t_{tx}[dBm] - 24.87\alpha - P^t_{intf}[dBm] \tag{3.4}$$

For a given *SNR* during transmission of one bit over an IEEE 802.15.4 link, the expected Bit Error probability (BEP) can be extracted using the Bit Error Rate (BER) model provided in the IEEE 802.15.4 standard (Annex E part 4.1.8) [2]. Based on this model, for O-QPSK modulation and Additive White Gaussian Noise (AWGN) channel model, BEP is calculated as:

$$BEP^t = \frac{8}{15} \times \frac{1}{16} \times \sum_{k=2}^{16} (-1)^k \binom{16}{k} e^{(20 \times SNR^t \times (\frac{1}{k} - 1))} \tag{3.5}$$

Using Equations (3.4) and (3.5), the BEP can be calculated for each bit of a packet, based on the interference power during its transmission and its transmission power. Accordingly, for a packet transmission started at time t with a length of L_{packet} (in bytes), we can compute the expected PRP as:

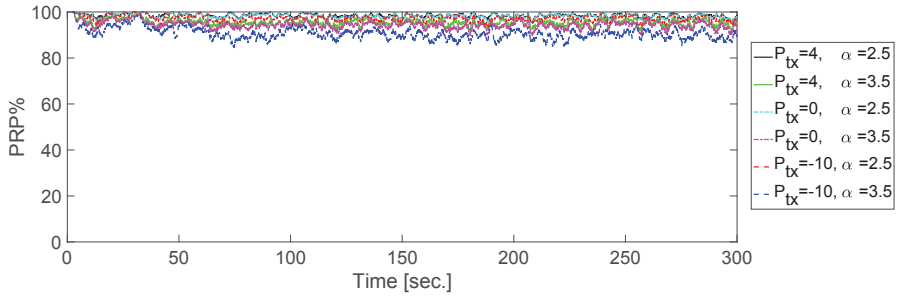
$$PRP^t = \prod_{k=0}^{(8 \times L_{packet})-1} (1 - BEP^{t+4k[\mu s]}) \tag{3.6}$$

We perform our simulations with $P_{tx} = 0$ dBm that is the default transmission power of the protocol, $P_{tx} = 4$ dBm that is the maximum transmission power of our ATMEL wireless motes, and also $P_{tx} = -10$ dBm to study the performance of very low-power communications. Because of the dynamism in a car (e.g., number of passengers and their position), the path-loss exponent is expected to be dynamic in an in-vehicle environment. We pick two values of $\alpha = 2.5$ and 3.5 for our simulations to investigate the effect of environment changes on the performance of the given wireless link. Therefore, we simulate the performance of the TSCH link for 6 different (P_{tx}, α) combinations under all different interference scenarios.

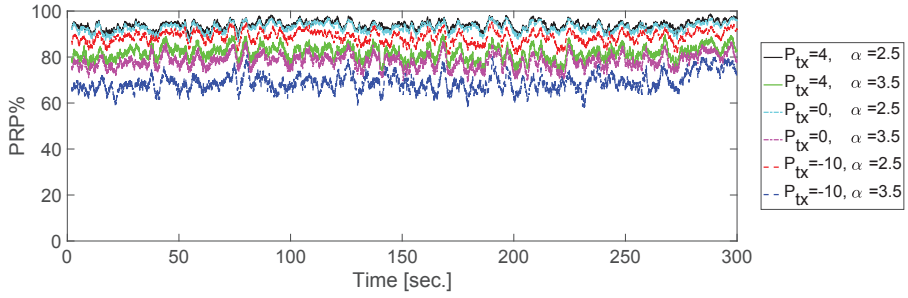
We implemented our simulation framework in Matlab according to the communication timings of the TSCH protocol. Time is divided into $10ms$ timeslots. After an offset at the beginning of each timeslot, we compute the BER for every bit using Equation (3.4) and the measured interference sample at that time on the operating channel. We consider a packet length of $L_{packet} = 133$ bytes which is the maximum physical layer packet length in the IEEE 802.15.4 protocol. By the start of the next timeslot, we hop to the next channel according to the TSCH hopping algorithm. We use all 16 available channels for the channel hopping.

In our interference measurements, the receiver sensitivity of the used devices was -90 dBm. This means that for all of the noise levels below this, the measured value is equal to -90 dBm. To alleviate the effect of this limit on the computed PRP, we replaced all the noise samples with a value of -90 dBm with -110 dBm in our data set. This guarantees that these samples have no effect on the computed PRP. Considering our worst case scenario with $P_{tx} = -10$ dBm and $\alpha = 3.5$, the PRP for P_{intf} = -110 dBm is 100%. Thus, we can be sure that our simulations only show the effect of existing interference.

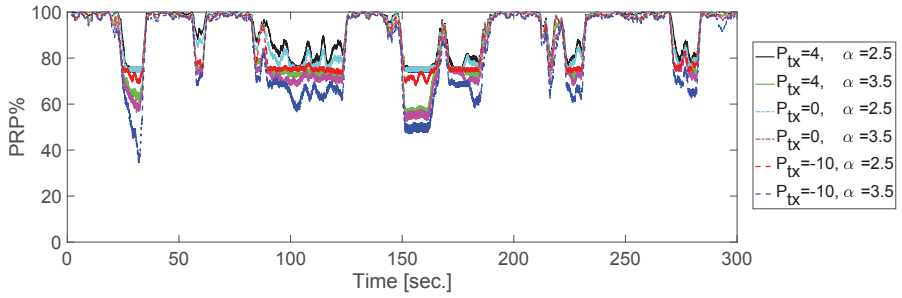
Figure 3.17 and Figure 3.18 show the simulated PRP over time for different scenarios. We used a moving average function with a window of 2 s (200 transmissions) to show the average PRP over time. This is an approximation of the PRR in real-world communications.



(a) Bluetooth mp3 streaming



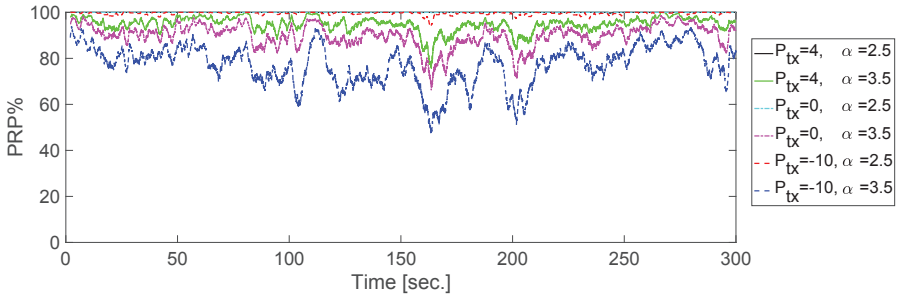
(b) Bluetooth file transfer



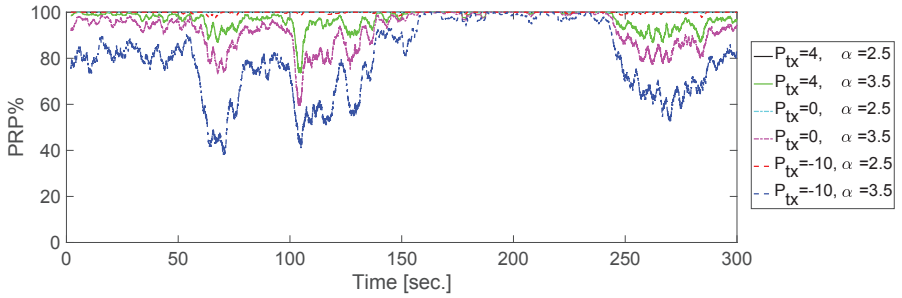
(c) Wi-Fi connection

Figure 3.17: Average PRP of TSCH communications over time for different in-car interference scenarios and communication parameters.

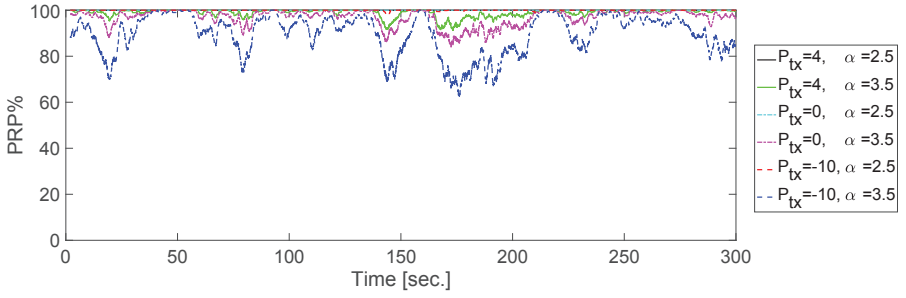
A general observation from the simulation results of different scenarios in Figure 3.17 and Figure 3.18 is that the path-loss exponent, which is a parameter of the link's environment, considerably affects the communications. For almost all different (P_{tx}, α) combinations in in-car interference scenarios (Figure 3.17), the PRP is affected by the interference. On the other hand, in out-of-car scenarios (Figure 3.18), the impact of interference on the PRP is considerably higher when $\alpha = 3.5$, but for other combinations of (P_{tx}, α) with $\alpha = 2.5$, interference has almost no effect on the PRP. The reason is that different in-car interferers usually produce high power interference on a set of channels



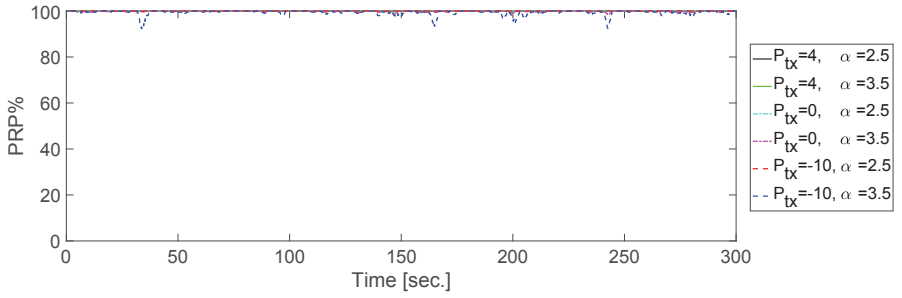
(a) Near apartments



(b) Downtown



(c) Suburb



(d) Highway

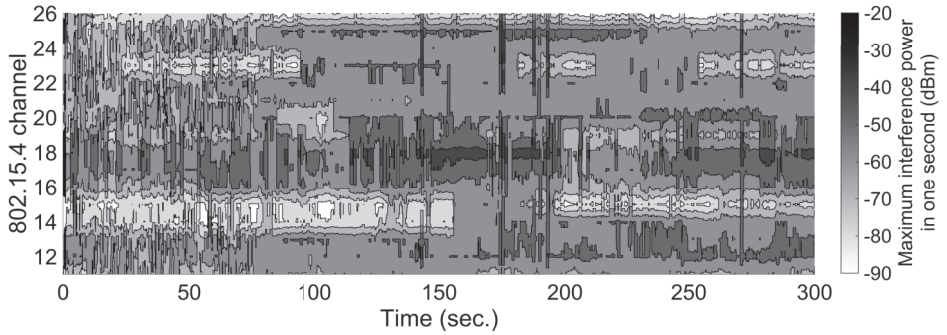
Figure 3.18: Average PRP of TSCH communications over time for different out-of-car interference scenarios and communication parameters.

(due to a low distance between interferer and the wireless node in the car) and less or even no interference on other channels (Figures 3.3, 3.7, and 3.10). Therefore, even for (P_{tx}, α) combinations with higher P_{tx} and lower path-loss exponents, the 802.15.4 link cannot overcome this high power interference on some channels and PRP will be decreased. For the out-of-car scenarios, the interference power is usually weaker (compared to in-car interference) but distributed over most of the channels (Figures 3.12-3.15). Thus, for $\alpha = 2.5$, this low-power interference has almost no effect on the PRP, as the SNR is high enough. However, for $\alpha = 3.5$, the SNR decreases and even low-power interference can affect the PRP. Because multiple channels in urban scenarios may experience interference at the same time (Figure 3.12-3.15), packet transmission may fail in a set of channels. Thus, PRP decreases considerably. This shows the importance of link selection in the network formation process, as links with higher quality (lower path-loss exponent) can be used for communications to mitigate the effect of out-of-car interference. Chapter 5 introduces a network formation technique to address this.

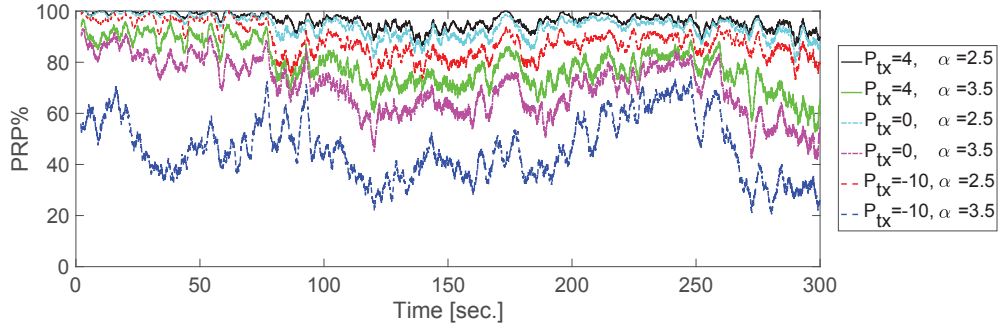
Figure 3.17 and Figure 3.18 also show that higher transmission powers may be a solution to increase the PRP. However, in the scenarios that the interference power is high (Figure 3.17(c)), the effect of using higher transmission powers on improving PRP is very low. On the other hand, WSNs are limited in power sources. Thus, transmission power of wireless nodes should be decreased as much as possible. Accordingly, other technique are required to increase the reliability of communications under external interference, without imposing more energy consumption to the network.

Figure 3.17 also shows that PRP in in-car scenarios is almost uniform over time. This is because the user applications in in-car interferers are usually invariant and running for a long time. Thus, wireless medium usage and generated interference is almost uniform over time. For the Wi-Fi scenario in which we transmitted some files in random intervals, the uniform behavior is visible for each file transfer (the periods with reduced PRP). Due to the movement of the interferers inside the car in this scenario, each file transfer leads to different interference power and thus different levels of PRP. For the out-of-car scenarios in Figure 3.18, because of the car movements, different interferers (with different user applications) may come into the communication range during time. Even in some periods, there may be no interferer in the communication range. Therefore, the effective interference and thus the PRP is very dynamic over time. This dynamism should also be supported by the employed interference mitigation techniques.

It should be considered that in real-world scenarios, out-of-car interference may be mixed with in-car interference, which may cause a bigger impact on the performance of an in-vehicle WSN. For example, a moving car in a downtown area may carry a mobile phone that is connected to the audio system of the car by Bluetooth to answer a phone call and at the same time a kid on rear seats may play an online video on a tablet which is connected to internet through a Wi-Fi hotspot link on a mobile phone. Since there can be a scenario with lots of interferers that block communications on all the channels, talking about the possible worst-case interference scenario is pointless. We consider the mentioned scenario as an example real-world scenario (named mixed scenario) with multiple sources of interference for an in-vehicle WSN. Figure 3.19 shows the captured interference of this scenario together with the simulation results of TSCH communications under this interference. As it can be seen, in such a scenario the PRP



(a) Interference behavior over 300 s using contour plot.



(b) Average PRP of TSCH communications over time.

Figure 3.19: Effect of the mixed interference scenario on the IEEE 802.15.4 channels.

of the TSCH protocol may go below 40% at some points of time. This happens more for links with lower qualities (higher path-loss exponents) and very low transmission powers. Accordingly, the network should be configured carefully; otherwise it may perform very poorly and not guarantee the Quality-of-Service (QoS) requirements. However, communications that use high transmission powers and links with higher qualities may still experience reliability problems. This requires other techniques to be employed to increase the performance of the network.

3.7 Summary

The cross-technology interference behavior in in-vehicle environments has been studied in this chapter. Real-world experiments for different scenarios are used for this purpose. A data set has been gathered as the result of these measurements that show the noise power on all the IEEE 802.15.4 channels in the 2.4 GHz band, for different automotive scenarios. Use of this data set provides more accurate analysis than lab or simulated data. This data set is public and available online [62]. The measurement results showed that interference affects all of the IEEE 802.15.4 channels, but distribution of the interference power is not uniform over time and channels.

The measurement data set was used as an input for a packet transmission model to study the behavior of the TSCH protocol under interference of different scenarios. The results show that interference of in-car sources leads to considerable probability of packet errors that is almost uniform over time (as in static environments). On the other hand, for out-of-car interference sources, the probability of packet errors can be highly dynamic over time. The simulation results show that even when using high transmission powers, interference affects communications of automotive WSNs. These observations show the necessity to use adaptive techniques to mitigate the dynamic interference in automotive environments. We propose an adaptive interference mitigation technique in Chapter 4. Simulation results also show the high impact of link qualities on the performance of the communications. Accordingly, we propose a network formation technique in Chapter 5 to improve the performance of an automotive TSCH WSN.

4

Interference Avoidance in Time-Slotted Channel Hopping

4.1 Overview

The IEEE 802.15.4 standard defines 16 frequency channels in the license-free 2.4 GHz ISM band. Depending on the MAC protocol used by a WSN, one or multiple channels may be used for communications. This frequency band is also used by other standards including IEEE 802.11 Wi-Fi [5] and IEEE 802.15.1 Bluetooth [1]. The common usage of this band leads to cross-technology interference and packet losses, especially for the WSNs that use low-power communication. This reduces the dependability of WSNs. Moreover, due to the vehicle movement in automotive WSNs, the cross-technology interference is dynamic over time. As shown in Chapter 3, this makes the performance of automotive WSNs unstable and unpredictable.

Using TSCH as the MAC protocol for automotive WSNs eliminates blocking of wireless links by hopping over multiple channels that are defined as a Hopping Sequence List (HSL). A HSL may include all or a subset of channels, to be determined by the upper layers in the protocol stack. Depending on the quality of channels in the HSL, a WSN may experience less or more interference over time. This can lead to bad performance of the TSCH MAC, if channels in the HSL are selected blindly.

The authors in [85] show that using static whitelisting to select the HSL in a static environment can increase the reliability of TSCH by restricting the used channels to the channels that are measured to be of good quality. As shown in Chapter 3, the automotive interference power on each channel is dynamic over time but stable over substantial periods of time. When the network conditions vary over time, an adaptive whitelisting is required to detect less noisy channels at the network run-time, and use them as the HSL. This increases the WSN dependability by preventing communication in noisy channels.

This chapter introduces ETSCH, an Enhanced version of the TSCH protocol that uses dynamic channel whitelisting. ETSCH uses a Non-Intrusive Channel-quality Estimation (NICE) technique to measure the quality of the IEEE 802.15.4 frequency channels. NICE

performs frequent channel samplings without applying any change to the protocol and does not require non-standard hardware. ETSCH also uses a secondary hopping sequence list to increase the transmission reliability of the Enhanced Beacon (EB) packets that are transmitted by the coordinator to set up the network. These techniques improve reliability of the TSCH protocol, by adaptively selecting a subset of low-noise channels for hopping.

ETSCH is centralized and is mainly done by the coordinator of the network. Considering that interference conditions at each node only affect packet receptions at that node, this technique is mainly preventing interference for packet receptions at the coordinator. In an automotive WSN, the interference conditions at different nodes of the network may be different on some channels. Accordingly, we propose a Distributed Channel Sensing (DCS) technique to estimate quality of the employed channels in all the nodes, and collect these quality assessments at the coordinator. The results are combined with the NICE results, to select a list of low-noise channels over the entire network area. In this chapter, we present DCS together with ETSCH, and call their combination ETSCH+DCS.

We use two sets of lab experiments with controlled interferers to evaluate ETSCH and ETSCH+DCS. In addition to the lab experiments, performance of the ETSCH technique is evaluated under realistic automotive scenarios, using the real-world interference data sets presented in Chapter 3. Experimental and simulation results show that ETSCH improves reliability of network communications, compared to basic TSCH and the state-of-the-art solutions. In some experimental scenarios, NICE by itself has been able to increase the average packet reception ratio by 22% and reduce the length of burst packet losses by half, compared to the plain TSCH protocol. Further experiments show that DCS can reduce the effect of hidden interference (which is not detectable by NICE) on the packet reception ratio of the affected links by 50%.

This chapter is based on two publications [72, 74] and is organized as follows. Section 4.2 gives an overview of adaptive multi-channel communications in WSNs. The detailed description of the proposed ETSCH+DCS technique is presented in Section 4.3. Evaluation setup and performance analysis are given in Section 4.4. Section 4.5 analyzes the energy consumption of the proposed techniques. Section 4.6 summarizes this chapter.

4.2 A Review of Channel Selection in Multi-Channel WSNs

The idea of multi-channel communications using channel hopping is used by a number of protocols and standards including WirelessHART [68], ISA100.11a [4], and TSCH. All of these protocols use a time-slotted approach to schedule network communications. At the start of each timeslot, which is synchronized in all network devices, each device hops to a new channel by use of a predefined Hopping Sequence pattern (HSL).

One of the enhancements to the channel hopping technique is to limit the used channels only to channels that are known to be of good quality. This technique is known as whitelisting; similar, blacklisting is the technique to skip using poor channels. WirelessHART [68] and ISA100.11a [4] protocols are designed for industrial applications and both use IEEE 802.15.4 radios in the 2.4 GHz ISM band. These protocols use a con-

figurable HSL at a global scope to control the channel hopping pattern. WirelessHART uses a channel blacklisting technique in which a user can add the channels that are affected by consistent interference to the blacklist. ISA100.11a uses the history of communications on each link and based on these statistics, the devices stop utilizing channels that are noisy for a particular period of time. While these protocols define some mechanisms to form and adapt HSL, the TSCH protocol leaves the channel selection task to the higher layers.

Watteyne et al. [85] show that use of a static whitelist/blacklist can improve performance of a TSCH network in nearly static wireless conditions. They try different channel whitelist sizes for their trace-based simulations and find out that a whitelist with size of 6 reduces the average Expected Transmission count (ETX) by 63%, compared to blind channel hopping using all 16 channels. ETX is the number of expected transmissions of a packet to successfully deliver it to the destination. Also, the authors conclude that the IEEE 802.15.4 channels 11, 15, 20, and 26 are much less interfered by IEEE 802.11 [5] (Wi-Fi). This is because Wi-Fi networks usually use the IEEE 802.11 channels 1, 6, and 11 which do not overlap with those four channels of the IEEE 802.15.4 standard.

For environments such as vehicles that experience a high level of interference dynamism, channel whitelisting/blacklisting needs to be done frequently to track interference over time. This requires use of an agile mechanism to perform channel quality estimation and HSL selection at run-time. Li et al. [46] propose an adaptive channel selection scheme based on the multi-arm bandit problem [33]. The selection of each channel is formulated as an independent process using packet transmission status (packet acknowledgement status) and Clear Channel Assessment (CCA) failures on that channel. In their proposed scheme, the channel selection is done on the transmitter side of each link. The channel list is transmitted to the coordinator by adding it to the information element of the TSCH packet. The coordinator broadcasts the new list of channels accordingly. Simulation results show that the algorithm is able to track existing interference on a channel in about 20 packet transmissions. This work does not specify the central channel selection method that is actually used. Furthermore, the packet transmission status that is used in this work is only available for ack-enabled transmissions, while real-time and multi-cast communications do not use acknowledgements. Finally, it is the local interference at the receiver(s) side that affects the communications and using packet transmission status does not show the interference at the transmitter side.

Gomes et al. [34] propose the Multi-hop And Blacklist-based Optimized TSCH protocol (MABO-TSCH). This technique uses the multi-armed bandit optimization for channel quality estimation at each node, using packet delivery ratio experienced by that node. The normal data packets and their acknowledgement are used to disseminate local blacklists to neighbors for negotiation process. Experimental results, with a 40-node indoor network, show that MABO-TSCH outperforms the default blind frequency hopping with a 23% higher throughput. This technique is suitable for large scale TSCH networks in which each node has a few neighbors and negotiation on a local blacklist does not require a lot of communication between neighbors. Therefore, it is not efficient to be used for the small and dense automotive networks that we target in this thesis.

Using solutions that only use the history of communications may not work well in dynamic environments. Because these solutions need prior communications on each

channel to gain enough knowledge about its condition. Moreover, it would be impossible to detect interference condition changes on a channel after it has been added to the blacklist. This will reduce the performance of these techniques when the interference conditions frequently change on each channel. Elsts et al. [26] propose an adaptive channel selection technique which uses a combination of central whitelisting and distributed blacklisting. The authors define two types of nodes, i.e. upstream and downstream nodes. The upstream node, which is the coordinator of the network, performs frequent RSSI samplings in all channels, and provides a channel whitelist without noisy channels. This HSL is broadcast to the network using EBs. Each downstream node extracts a blacklist based on the packet delivery ratio of its transmitted packets. Every time a downstream node wants to send data, it uses the channels that are in the central whitelist but not in the local blacklist. This technique uses the whole Tx offset of timeslots for RSSI samplings and therefore it may detect internal interference in the network as external interference. Also, the RSSI procedure on different channels is not specified. This technique may also leads to an empty or very small channel hopping list at some nodes, if local blacklist at each node and the central whitelist share a lot of common channels. This technique also makes the number of available channel offsets unpredictable.

To cope with dynamic wireless medium conditions, Du et al. [23] propose the Adaptive Time-Slotted Channel Hopping (ATSCH), a dynamic whitelisting/blacklisting mechanism using hardware-based Energy Detections (EDs). ATSCH works on top of the TSCH protocol and reserves two timeslots in each TSCH slotframe to perform energy samplings on the operating channel of these timeslots. There will be no communications in these timeslots; therefore the gathered values of energy samplings can be considered as noise levels on those channels. These sampling results are used by each node to assign a quality factor to each channel and periodically add the low-quality channels to its local blacklist. Nodes share their local blacklists with neighbors so that the sender and receiver of a link use the share of their blacklists as HSL. This may cause problems for parallel TSCH communications, as different links may use different HSLs that result in using the same channel for different channel offsets at the same timeslot. For a small and dense automotive network, a global HSL can be defined by the Personal Area Network (PAN) coordinator (central coordinator of the network) to solve this problem. This HSL can be periodically broadcast to all nodes using EBs.

Our proposed ETSCH+DCS mechanism uses hardware-based EDs together with transmission logs to measure the quality of channels and select the best subset of channels as HSL. Although ATSCH and ETSCH both use the same philosophy of hardware-based channel sampling methods, there are several advances in ETSCH.

- ATSCH reserves two timeslots of a TSCH slotframe which results in a reduced throughput. ETSCH does not use transmission parts of timeslots, and thus does not reduce the capacity of the network and requires no change to the TSCH schedule.
- The rate of sampling in ATSCH is two samples per slotframe and is directly affected by the size of the slotframe. In contrast, ETSCH introduces the NICE technique to perform energy samplings at least two times per timeslot. It thus has a sampling rate that is at least L_{SF} times higher than that of ATSCH. This makes ETSCH to perform better in highly dynamic wireless conditions.

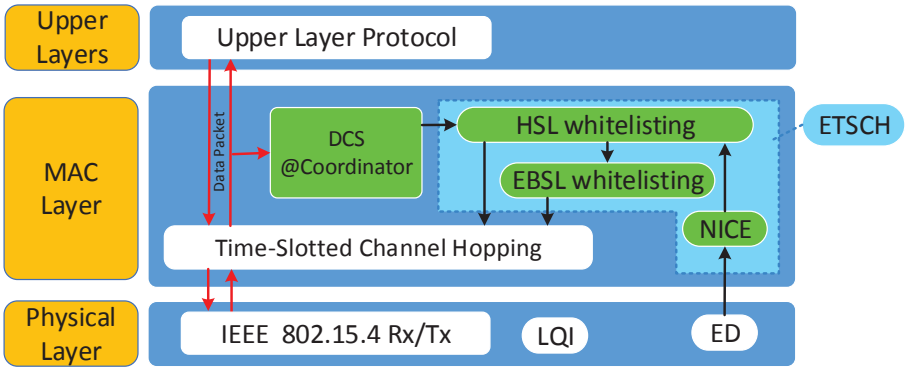


Figure 4.1: ETSCH+DCS components in the coordinator node

- ATSCH uses all sixteen channels to broadcast EB periodically (containing the HSL) which may result in EB losses and synchronization loss between nodes. We propose a new method to broadcast EBs in a TSCH network using a secondary HSL. This technique uses a small and less dynamic hopping sequence list which contains the best quality channels. By using this secondary hopping list for EB transmissions, the probability of EB losses is reduced. Jeon et al. [43] propose a technique to adaptively change the frequency of EDs in ETSCH based on the interference dynamics. This reduces the energy consumption of ETSCH when power is a constraint at the coordinator node.

We also employ a distributed channel quality estimation technique called DCS, using packet reception and CCA logs, to detect interference at the position of non-coordinator nodes of the network. Thus, it considers noise at the point of other nodes of the network, which may be invisible at the coordinator node. This is while other blacklisting/whitelisting techniques that use history of communications at the coordinator (such as [46]) cannot detect and mitigate the existing interference at non-coordinator nodes of the network. Furthermore, ETSCH uses hardware-based samples to update the assigned quality to a blacklisted channel. In communication-based channel quality estimation techniques, it is impossible to detect channel condition changes after a channel is blacklisted.

4.3 Enhanced Time-Slotted Channel Hopping with Distributed Channel Sensing

In this section we describe all the components of ETSCH together with the DCS technique in details. ETSCH components include NICE, HSL whitelisting, and EB hopping Sequence List (EBSL) whitelisting. We start with a brief overview of the functionality of all components and their relation, and then we describe each component in detail.

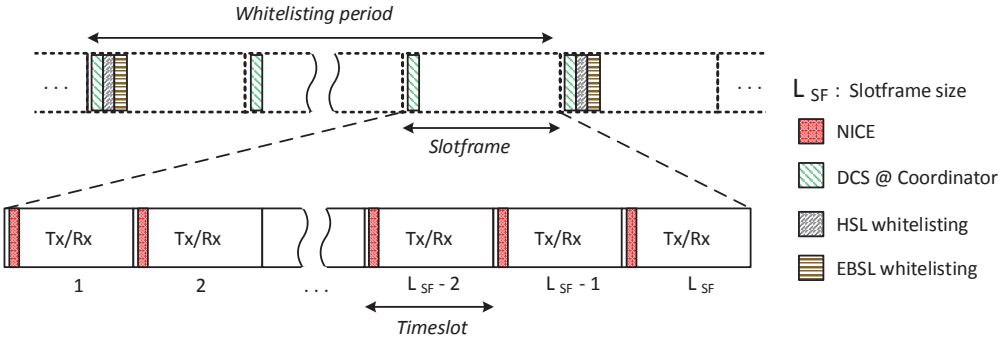


Figure 4.2: Occurrence of ETSCH components within the TSCH slotframe and timeslot structure in the coordinator node

4.3.1 Overview

The basic idea of ETSCH is to adaptively select a subset of low-noise channels called the *whitelist*, and use it as an input for the channel hopping algorithm. Centralized whitelisting performs well for automotive WSNs with small area, in which all the nodes are in the communication range of the coordinator. ETSCH adds three components to the basic TSCH protocol at the coordinator node. Figure 4.1 shows the placement of these techniques together with DCS technique within the protocol stack at the coordinator node. Figure 4.2 shows their occurrence within the TSCH slotframe and timeslot structure. NICE runs every timeslot, DCS@coordinator runs every slotframe, and HSL and EBSL whitelisting techniques run every whitelisting period. It should be considered that all of these components are very lightweight and run in the idle part of timeslots, thus their execution has no impact on the protocol functionality. Algorithm 1 shows the process of each of these components at the coordinator. Each wireless node in the network has a DCS component to sense the channel conditions at that node and report them to the coordinator.

As Figure 4.1 shows, NICE runs in parallel with TSCH on the MAC layer of the coordinator to extract the quality of all available channels. NICE uses the EDs introduced in the protocol to measure the quality of each frequency channel. The first part of Algorithm 1 shows the process of the NICE technique (lines 1 to 8). NICE uses the *silent period* in every coordinator timeslot to perform as many EDs on different channels as possible. Based on the timeslot diagram of the TSCH protocol and considering the fact that the coordinator is the time source of the network, this silent period is only available at the coordinator of the network. EDs are performed on successive channels and after 16 EDs, all the channels are sampled. The result of each ED is used to update the assigned Channel-Quality Estimation (CQE) to that channel. Figure 4.2 shows the occurrence of NICE in the silent part of each timeslot.

NICE provides centralized interference detection for ETSCH. It does not address the challenge that an interferer that is hidden from the coordinator may cause interference for some of the nodes in the network on some channels. To address this problem, it is necessary to employ a distributed channel quality estimation technique together with

ALGORITHM 1: ETSCH+DCS components

Data:
CQE []: an array to store Channel Quality Estimation results of all channels
HSL []: an array to store the main Hopping Sequence List, to be used by TSCH
EBSL []: an array to store Enhanced Beacon hopping Sequence List, to be used by TSCH

```
1  NICE (CQE [])
2      every timeslot do
           /* This algorithm allows repetitive channel samplings in each silent period */
3          while it is the silent period do
4              ch ← (((ch − 11) + 1)%16) + 11; /* select successive channels in the range of 11
                   to 26                                                                    */
5              CQE [ch] ← EWMAFilter(ED (ch)); /* see EWMA Filter in Equation (4.5)        */
6          end
7      end
8  end
9  DCS@Coordinator(HSL [], CQE [])
       Input: PKT(node_id): Data packet received from device node_id
       Data: CC []: an array to store Channel Conditions that are received from different nodes
10     foreach PKT(node_id) received by coordinator do
11         CC [node_id] ← extract channels_condition field from PKT(node_id);
12     end
13     every Slotframe period do
14         foreach ch in HSL do
15             CC_avrg [ch] ← average of all channel conditions recorded in CC [] for channel ch;
16             CQE [ch] ← EWMAFilter(CC_avrg [ch]); /* see EWMA Filter in Equation (4.13)
                   */
17         end
18         Clear CC[];
19     end
20 end
21 HSL_whitelisting(CQE []), |HSL|)
       Input: |HSL|: size of Hopping Sequence List
       Output: HSL []
22     every whitelisting period do
23         HSL_sorted []← Ascending sort of channels based on CQE [];
24         HSL [] ← HSL_sorted [1 to |HSL|];
25     end
26 end
27 EBSL_whitelisting(EBSL [], HSL [], k)
       Input: k: the EBSL entry that was used for the last EB transmission
       Output: EBSL []
28     every whitelisting period do
29         if EBSL [k] ∉ HSL [0 to 3] ∧ EBSL [k] ≠ 26 then
30             m = min{h | 0 ≤ h ≤ 3 ∧ HSL [h] ∉ EBSL}; /* Find the channel which is not in
                   the EBSL and has the best quality                                      */
31             EBSL [k] ← HSL [m];
32         end
33         k ← the EBSL entry which is used for EB transmission in this timeslot;
34     end
35 end
```

Table 4.1: Default value of some of the timeslot offsets defined in the IEEE 802.15.4 protocol [7].

Attribute	default value (μs)
macTsTimeslotLength	10000
macTsCcaOffset	1800
macTsTxOffset	2120
macTsRxOffset	1020
macTsRxWait	2200

NICE. As we show in the following, the mentioned silent periods are only available at the coordinator of the network. Thus, NICE cannot be used in other nodes to perform EDs to extract the quality of channels. We use CCA and packet reception status as two channel quality estimators in other nodes of the network. By using these techniques, each node declares each channel as noisy or noiseless and includes the results in the data packets that it sends to the coordinator. Because the status of each channel is a binary value, transmitting the status of all 16 channels leads only to a 2 Byte overhead. At the coordinator, the channels' status field is extracted from all the incoming packets and is collected by the "Distributed Sensing @Coordinator" component (shown in Figure 4.1). As Figure 4.2 depicts, at the beginning of each slotframe, the collected data is analyzed. Accordingly, the assigned CQE to each channel is updated. The process of this technique at the coordinator node is shown in lines 9 to 20 of Algorithm 1. This DCS technique is presented in detail in Section 4.3.3.

The output of NICE and the DCS technique, as a single CQE array, is used periodically by the HSL whitelisting component which runs in the MAC layer at the coordinator node (lines 21 to 35 of Algorithm 1). The HSL whitelisting component selects a subset of best quality channels for TSCH, based on the observed wireless conditions. According to the results of HSL whitelisting, EBSL whitelisting selects a subset of best channels, in a less dynamic way compared to the HSL whitelisting, to be used for EB transmissions. The HSL and EBSL whitelisting components are performed periodically once every a couple of slotframes, so called whitelisting period. Figure 4.2 shows that these two components are executed at the beginning of the first slotframe of each whitelisting period, after execution of the DCS component. The results of HSL and EBSL whitelisting is included into the EB packets and broadcast to the other nodes in the first slot of each slotframe.

4.3.2 Non-Intrusive Channel-quality Estimation

To perform an ED in a frequency channel to estimate its noise level, there should be no transmissions in the network itself during that measurement. We propose NICE to perform the EDs on different frequency channels at coordinator and without any bandwidth cost to the protocol.

As discussed in Chapter 2, to compensate an amount of timeslot phase differences caused by clock drifts, TSCH defines some timeslot offsets. To extract the maximum allowed phase difference for default values of these offsets (given in Table 4.1), we investigate different cases. As illustrated in Figure 4.3(a), if a receiver starts its timeslot $T_{forward}$ = 1100 μs earlier than the coordinator, it still can receive the packet from the

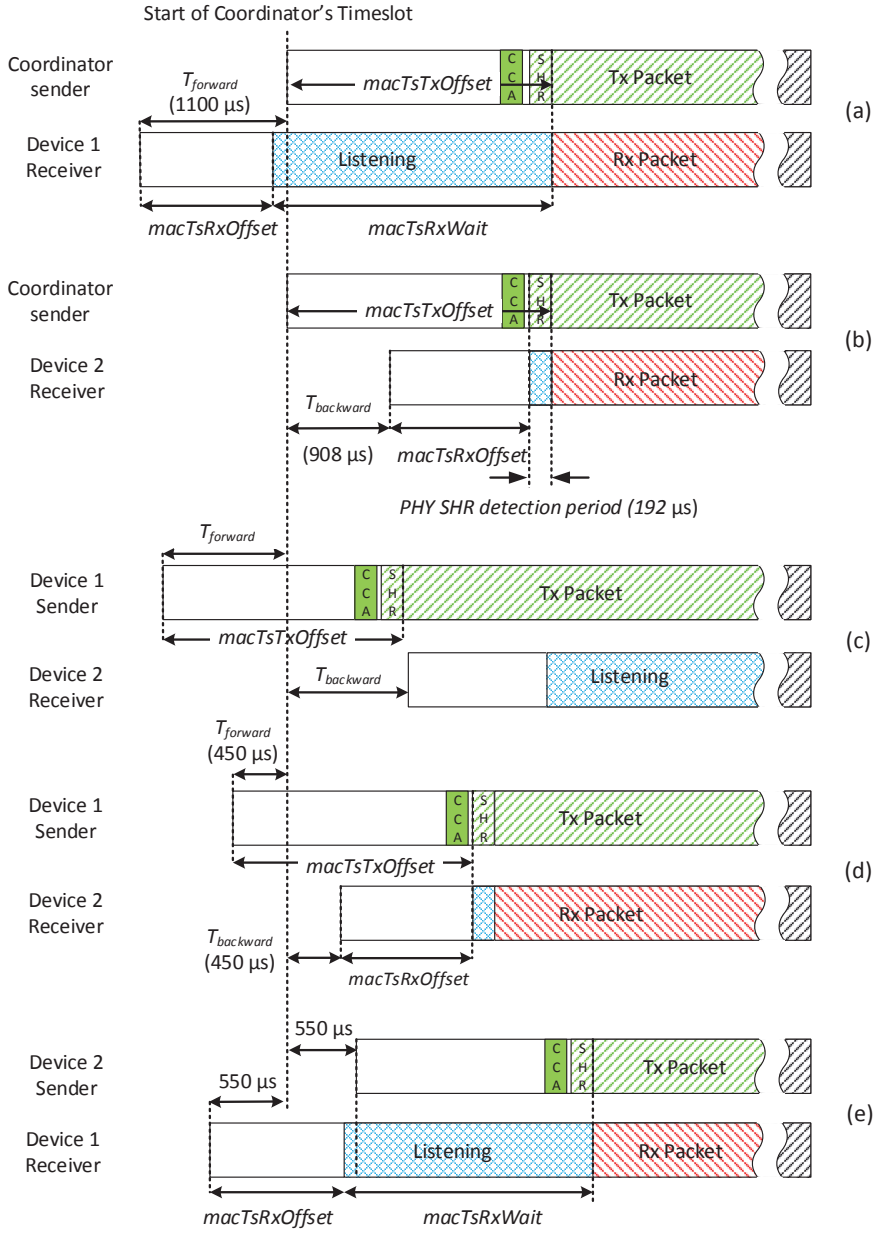


Figure 4.3: Pair-wise communications in the case of timeslot phase difference; (a) Device 1 starts $T_{forward} = 1100\ \mu s$ ahead the coordinator, (b) Device 2 starts $T_{backward} = 908\ \mu s$ later than the coordinator, (c) Communication of device 1 with $T_{forward} = 1100\ \mu s$ and device 2 with $T_{backward} = 1100\ \mu s$ fails, (d) Successful communication of devices 1 and 2 with $T_{forward} = T_{backward} = 450\ \mu s$, considering the required period for PHY Synchronization Header (SHR) detection at the receiver, (e) Successful communication of devices 1 and 2 with $T_{forward} = T_{backward} = 550\ \mu s$

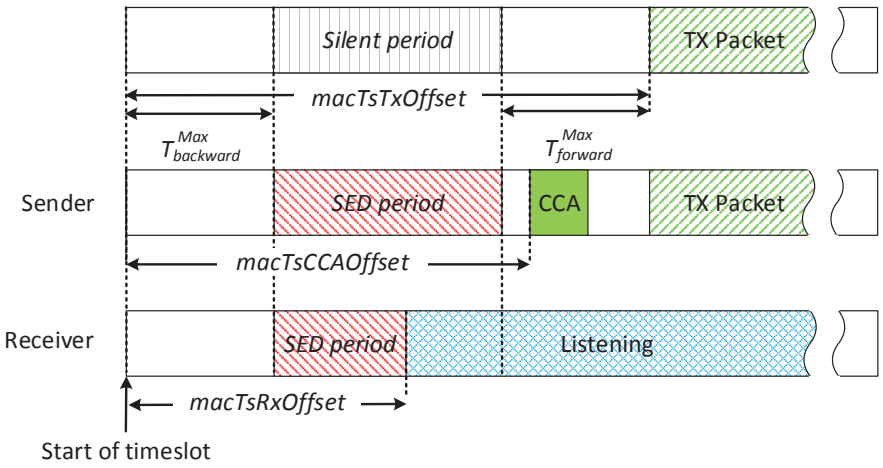


Figure 4.4: Available time for Silent Energy Detection (SED) when coordinator is a sender or a receiver

coordinator. Also, if a receiver starts its timeslot $T_{backward} \approx 900\ \mu s$ later than the coordinator, the two nodes can still communicate (Figure 4.3(b)). This value is extracted by considering the required time for PHY SHR detection at receiver, which should happen before receiving the MAC packet. Considering the fact that the coordinator of a wireless network is the main source of synchronization, there is no chance for a device that starts $T_{forward} = 1100\ \mu s$ before the coordinator to communicate with a device that starts $T_{backward} = 1100\ \mu s$ after the coordinator (Figure 4.3(c)). To enable bidirectional transmission between each pair of nodes in the network, as shown in Figure 4.3(d) and (e), the forward and backward timeslot phase differences should be less than $T^{Max}_{backward} = T^{Max}_{forward} = 450\ \mu s$. It means that the employed timeslot synchronization method should guarantee the synchronization loss to be less than these values in order to have a connected mesh network.

Each device may start its timeslot at maximum $T^{Max}_{forward}$ earlier or $T^{Max}_{backward}$ later than the coordinator. Therefore, from the coordinator perspective, for a $T^{Max}_{backward}$ time period at the start of each timeslot, there is a possibility of packet transmissions by some nodes in the previous timeslot. Also for a $T^{Max}_{forward}$ time period before *macTsTxOffset*, there is the possibility that some nodes start packet transmissions ahead of the coordinator. Considering these possibilities, there will be no packet transmission expected in the network for a T_{silent} period (Figure 4.4) that can be computed by Equation (4.1). For the timing defaults of the TSCH protocol, this value is $T_{silent} = 1220\ \mu s$.

$$T_{silent} = macTsTxOffset - T^{Max}_{backward} - T^{Max}_{forward} \tag{4.1}$$

The reference time to declare the silent period is the start of the coordinator's timeslot. Thus the coordinator of the network knows the exact start and end point of this period over time. On the other hand, because of the allowed timeslot phase difference

between network nodes and the coordinator, non-coordinator nodes cannot have an exact estimation about the start of the coordinator's timeslot. It makes it impossible to determine the silent period at those nodes. To perform an ED in a network and have an estimation about the noise level of the channels, there should be no transmissions in the network during these measurements. Therefore, we use this silent period in each timeslot to perform EDs, only on the coordinator device.

A wireless device can be receiver, transmitter, or an idle node during a timeslot. According to this, EDs during the silent period in the coordinator device can be divided into three types. If the coordinator is a receiver, it should finish the ED process within the *macTsRxOffset* period. The overlap of this period and the silent period can be used for the ED process. When the coordinator is a transmitter, this period will be the overlap of *macTsCCAOffset* and the silent period. The whole silent period can be used for performing EDs when the coordinator has no *Rx/Tx* task. Figure 4.4 shows these Silent Energy Detection (SED) periods. The available ED duration for each type of timeslot transmission can be computed as Equations (4.2), (4.3), and (4.4).

$$T_{SED}^{Rx} = \min(macTsRxOffset, macTsTxOffset - T_{forward}^{Max}) - T_{backward}^{Max} \tag{4.2}$$

$$T_{SED}^{Tx} = \min(macTsCCAOffset, macTsTxOffset - T_{forward}^{Max}) - T_{backward}^{Max} \tag{4.3}$$

$$T_{SED}^{Idle} = T_{silent} \tag{4.4}$$

According to the TSCH protocol defaults, these ED periods will be $T_{SED}^{Rx} = 570\ \mu s$, $T_{SED}^{Tx} = T_{SED}^{Idle} = 1220\ \mu s$. Each ED takes 8 symbol periods and is the mean of 128 consequent measurements of the signal strength, each lasting for $1\ \mu s$. To hop to the desired channel for performing EDs and also get the ED measurements from hardware and performing the quality estimation evaluations, we assume this time to be more than twice as high, namely $T_{ED} = 280\ \mu s$ (the value observed in our experiments is less). Therefore, a coordinator can perform two EDs in receiving timeslots and three to four EDs in transmitting and idle timeslots (we consider three EDs for our experiments). Each ED is done in the channel next to the channel for which the prior ED was done.

Considering the default length of a timeslot as 10 ms and the least number of possible EDs per timeslot to be two, the minimum channel sampling rate will be 200 samples per second. Considering the 16 available channels of IEEE 802.15.4 at the 2.4GHz band, each channel will be sampled about 12 times per second. This sampling rate is independent of the slotframe size. Furthermore, NICE imposes no throughput cost to the protocol, i.e., it is non-intrusive.

Results of the EDs (higher values shows higher noise on the channel) are used to assign CQE values to each of the channels. To compute a stable estimate of the channel quality, as the ED measurements may fluctuate, we use an exponential smoothing technique [32]. This technique is also used by ATSCH to compute CQE values. Every time a new ED is done in a channel, a new CQE is calculated for it using Equation (4.5).

$$CQE_{\tau}(ch) = \alpha(ED^{max} - ED_{\tau}(ch)) + (1-\alpha)CQE_{\tau-1}(ch) \tag{4.5}$$

where $ED_\tau(ch)$ is the new ED sample of channel ch and $CQE_{\tau-1}(ch)$ is the last computed CQE for that channel. Higher values of $ED_\tau(ch)$ shows higher noise level. Thus, we subtract this value from the maximum ED value (ED^{max}) so that lower noise on a channel (i.e., low $ED_\tau(ch)$ values) leads to higher $CQE_\tau(ch)$ values. Coefficient α, with $0 < \alpha \leq 1$, is the smoothing coefficient that controls the effect of new ED samples on the CQE_τ. By selecting small values of α, we obtain stable estimation of channel quality values. Very small values impose a delay in detection of changes in the quality of a channel though.

4.3.3 Distributed Channel Sensing

Due to synchronization loss caused by clock drifts, it is impossible to determine the silent period at non-coordinator nodes. Thus, we cannot use NICE to extract quality of channels in non-coordinator nodes. However a third hop interferer, which is hidden from the coordinator, may generate interference for some of the other nodes. Therefore, we employ a distributed channel quality estimation technique called DCS to work together with NICE as the interference detection block of ETSCH.

The overhead of such a sensing technique should be taken into consideration. Using channel EDs at non-coordinator nodes (as ATSCH [23]), leads to extra power consumption that is a negative point in battery-powered wireless nodes. We take advantage of CCA and packet reception status, which are already available and defined in the protocol, as two parameters representing conditions of each communicating channel. All of these parameters give an estimation about the channel quality at the point of the measuring/receiving node, which provides enough information about the existence of interference at each node. The only limitation of using these parameters is that they only give information about the condition of the channels that are being used for communications. In other word, the condition of blacklisted channels is not extractable using these parameters. However, this is not an issue in ETSCH since when a channel is detected as a bad quality channel and is blacklisted, NICE still measures its quality and updates the assigned CQE to it. Therefore, if the coordinator realizes that the quality of a blacklisted channel is better than a used channel, there is a chance for that channel to be used for communications again.

The CCA is already defined in the IEEE 802.15.4 standard [7], as a part of the transmission diagram in transmitter nodes. As discussed in Chapter 2, CCA can be done by detecting energy above a threshold (using ED to detect interference from wireless devices out of the network) or carrier sense detection (detect interference from sensor nodes in the network or neighbor networks, with the same modulation) or both. Normally, for a WSN that experiences interference from different sources, those modes with detecting energy above a threshold work better. This is because carrier sense detection may miss most of the interference, due to use of different modulations by other networks, while energy detection can detect interference caused by any type of modulation. By using these operation modes, for most of the IEEE 802.15.4-based sensor nodes, it is also possible to read the ED value from the radio after a CCA. Here we only consider the CCA result, which is a *Channel_idle* or *Channel_busy*, but the attached ED value can also be taken into account to calculate the quality of the channel. Every time a new CCA is done in node i (before transmitting a packet) in a dedicated slot, a new Channel Quality $CQ_{i,\tau}$ is calculated for that channel. We skip CCA in Tx slots with link

option *shared* enabled. In these slots several nodes may try to transmit, and CCA is used mainly to detect and avoid internal collisions. As we want to detect the presence of external interferences, we only use the results of CCA performed in the dedicated slots in which only one node may transmit.

As for NICE, we use exponential smoothing to calculate the new channel quality $CQ_{i,\tau}$ based on its previous value $CQ_{i,\tau-1}$. Because CCA results are boolean values, we use maximum and minimum values of $CQ_{i,\tau}$ as the input data for quality calculations. Equation (4.6) shows the used formula.

$$CQ_{i,\tau}(ch) = \begin{cases} \alpha' CQ^{Max} + (1-\alpha')CQ_{i,\tau-1}(ch) & \text{if CCA returns } \textit{Channel\_idle} \\ \beta' CQ^{Min} + (1-\beta')CQ_{i,\tau-1}(ch) & \text{if CCA returns } \textit{Channel\_busy} \end{cases} \quad (4.6)$$

Here, CQ^{Max} and CQ^{Min} are two constants which are set to upper and lower values of $CQ_{i,\tau}$, i.e. 255 and 0, respectively. Coefficient α' and β', with $0 \leq \alpha', \beta' \leq 1$, controls the effect of a *Channel_idle* or *Channel_busy* on the computed channel quality, respectively. We use two different values to be able to give different weights to positive or negative CCA results. Lower values of each coefficient reduce the effect of a new CCA status on $CQ_{i,\tau}$ and thus makes the quality estimation more stable over time. On the other hand, higher values makes $CQ_{i,\tau}$ more adaptive to new conditions. The relation between these two coefficients shows the expected reliability of communications on each channel. When $\alpha' < \beta'$, $CQ_{i,\tau}$ is more sensitive to *channel_busy* samples compared to *channel_idle* samples. Thus, in order to have higher $CQ_{i,\tau}$ values for a channel, the probability of *channel_busy* occurrence should be lower than *channel_idle* occurrence. This guarantees a high communication reliability for channels with a high $CQ_{i,\tau}$. On the other hand, when $\alpha' > \beta'$, even channels with higher occurrence of *channel_busy* compared to *channel_idle* may get high $CQ_{i,\tau}$ values. This leads to possible use of unreliable channels for communications when channels with high $CQ_{i,\tau}$ values are used for communications.

Packet reception status shows the success or failure of an expected packet transmission at the receiver side, without considering acknowledgement of that packet. It is different from the packet delivery status, which can only be extracted if acknowledgement is enabled. To use the packet reception status in our quality estimation technique, we should be sure that in each dedicated timeslot, there is at least one packet transmission. Otherwise, a failure in packet reception may mean a failure due to interference or skipping transmission by the transmitter. We refer a dedicated timeslot to a timeslot with link option *Tx* enabled and link option *Shared* disabled. We assume that even if there is no data to be transmitted by a source node on a dedicated Tx timeslot, it should send a keep alive or dummy packet to the assigned destination. As we exclude the Tx slots with the *shared* option enabled (these slot can be used for overprovision, association commands and other sudden traffic), this technique does not affect the functionality of shared timeslots. It should be considered that in a real schedule the number of dedicated slots should be in line with the required bandwidth by the application. Thus, if a mature slot scheduling mechanism is in place, all dedicated timeslots contain real data transmission in most slotframes. Therefore, transmission of the dummy packets are rarely required.

Considering the default timeslot timings that are specified in the standard, we claim that if dummy packets contain no payload, and also use the shortest possible MAC header, they apply no power overhead to the network. Based on the IEEE 802.15.4 standard [7], the PHY header is 6 Bytes and the minimum MAC header which contains frame control, PAN identifier, destination address, source address, and frame check sum fields is 9 bytes. As defined in the IEEE 802.15.4 standard [7], the receiver nodes in a dedicated timeslot starts listening at $macTsRxOffset = 1020\mu s$ and waits for $macTsRxWait = 2200\mu s$ to receive the frame preamble. Considering the transmission offset which is $macTsTxOffset = 2120\mu s$, the receiver continues listening for extra $(macTsRxOffset + macTsRxWait) - macTsTxOffset = 1100\ \mu s$ from the expected Rx start time, to receive the packet. This is due to the possibility of synchronization loss between the transmitter and receiver. Therefore, if a transmitter has no data to transmit in a dedicated timeslot and skips the transmission, the intended receivers waste a lot of power in the listening phase.

We estimate the accumulative energy consumption of receiver and transmitter nodes in both cases of transmitting and not transmitting a dummy packet. If T_{Rx} represents the duration of the listening period at the receiver and T_{Tx} represents the duration of a packet transmission, the overall energy consumption (E) of each transmission is

$$E = \big((I_{Rx} T_{Rx}) + (I_{Tx} T_{Tx})\big) \times V_{cc} \tag{4.7}$$

where I_{Rx} and I_{Tx} stand for the radio transceiver current in receive and transmit modes, respectively. Based on the symbol timing definitions in the standard, transmitting each Byte takes 32 μs. Thus transmitting a dummy packet with PHY packet size of 15 bytes takes $15 \times 32 = 480\mu s$. By considering the average synchronization loss between transmitter and receiver as $0s$, the receiver keeps the radio ON for only T_{Tx} time after start of transmission. Considering the fact that for low-power wireless sensor nodes, the radio current in Rx mode is almost equal or even higher than the radio current in Tx mode [81](e.g. Atmel ATmega256RFR2 wireless microcontroller [50]), the overall energy that is consumed for transmission of a dummy packet can be computed as

$$\begin{aligned} E_{dummy} &= \big((I_{Rx} \times (480[\mu s] + (macTsTxOffset - macTsRxOffset)) + (I_{Tx} \times 480[\mu s])\big) \times V_{cc} \\ &\approx \big(I_{Rx} \times (480 + 1100 + 480)[\mu s]\big) \times V_{cc} \\ &= 2060[\mu s] \times I_{Rx} \times V_{cc} \end{aligned} \tag{4.8}$$

If the transmitter skips packet transmission at a dedicated timeslot, the overall energy consumption will be

$$\begin{aligned} E_{No\text{-}Tx} &= \big((I_{Rx} \times macTsRxWait) + (I_{Tx} \times 0)\big) \times V_{cc} \\ &= 2200[\mu s] \times I_{Rx} \times V_{cc} \end{aligned} \tag{4.9}$$

The results of Equations (4.8) and (4.9) show that the overall energy consumption (sum of energy consumed by transmitter and receiver) when we transfer a short dummy packet is less than the case that no transmission takes place. This energy consumption overhead is even more visible when there are multiple receivers listening to receive a packet from a single transmitter (multicast or broadcast transmissions) that transmits nothing.

These energy consumption calculations are for default timeslot timings defined in the IEEE 802.15.4 [7] standard document. For other timeslot timings, transmitting a dummy packet saves energy if $E_{dummy} < E_{No\text{-}Tx}$. The relation between timeslot offsets is defined in the protocol document as $macTsRxOffset + macTsRxWait/2 = macTsTxOffset$. Having this and Equations (4.8) and (4.9), transmitting a dummy packet saves energy when $(macTsTxOffset - macTsRxOffset) > 960$. Actually, this technique also improves the channel quality estimation accuracy of ETSCH and helps preventing packet failures caused by using noisy channels. This can lead to energy saving gain even for shorter timeslot offsets. We suggest transmitting dummy packets when timeslot offsets meet the mentioned condition. Otherwise, it is the user choice to skip packet reception status to be used in DCS technique.

Based on the above discussion, when a timeslot is dedicated to a node for packet transmission and there is no data to transmit, we assume that the node transmits a small dummy packet instead. Thus, if a receiver node does not receive a packet in a dedicated timeslot, or there is a CRC error when it receives a packet, this condition can be considered as a *transmission_failure*. Otherwise, the packet reception status will be a *transmission_success*.

After each (expected) packet reception, we use the packet reception status to update the assigned quality to the channel that is used for packet reception. It is also possible to use the Link Quality Indicator (LQI) result which is attached to the received packet by the radio. Actually, LQI shows the communication quality between two nodes on a channel, not only the quality of the channel. Considering this, we only take the packet reception status into consideration. Every time a new packet reception is done in a channel at node i, a new Channel Quality $CQ_{i,\tau}$ is calculated for that channel using Equation (4.10).

$$CQ_{i,\tau}(ch) = \begin{cases} \alpha' CQ^{Max} + (1-\alpha') CQ_{i,\tau-1}(ch) & \text{if } \textit{packet reception succeeds} \\ \beta' CQ^{Min} + (1-\beta') CQ_{i,\tau-1}(ch) & \text{if } \textit{packet reception fails} \end{cases} \tag{4.10}$$

Here we use the same technique and coefficients that are used for $CQ_{i,\tau}$ calculation in Equation (4.6). This is because both CCA and packet reception status show the success or failure in using the channel and thus they should affect the $CQ_{i,\tau}$ in the same way. It is also possible to use different coefficients for Equation (4.6) and Equation (4.10), as one of them may show the channel condition better than the other according to the type of interference.

Using the described technique, when a channel is not used in the HSL and there is no transmission on it, its $CQ_{i,\tau}$ does not change at all. When this channel is added again to the HSL, its channel quality value on each node refers to the past and may affect new channel quality calculations. Therefore, on every node i we reset the $CQ_{i,\tau}$ value of each newly used channel in the HSL to a predefined constant value CQ^{init}. This initial value should give a chance to the newly added channels to be used for communications. If the quality of a newly added channel is still bad, it should be removed from the HSL after a few transmissions.

Because the coordinator is responsible for defining the HSL and distributing it, the results of the DCS technique at all the nodes should be gathered by the PAN coordinator. We attach this data to the normal data packets that are transmitted from end nodes

to the PAN coordinator (unicast and broadcast packets). Transmitting the calculated channel quality of all 16 channels of each node to the PAN coordinator results in a high throughput overhead for the network. To reduce this overhead, in each node i we use a threshold-based whitelisting and apply it on $CQ_{i,\tau}$. By using Equation (4.11), the channels that have a quality greater than a predefined threshold θ, are selected as good channels to be reported to the coordinator.

$$CC_i(ch) = \begin{cases} 0 & \text{if } CQ_{i,\tau}(ch) < \theta \\ 1 & \text{if } CQ_{i,\tau}(ch) \geq \theta \end{cases} \tag{4.11}$$

Here, $CC_i(ch)$ is the whitelisting result (channel condition) of channel ch at node i. Threshold θ should be lower or equal to CQ^{init}. This is because we initially want to consider a newly added channel ch to the HSL, with $CQ_{i,\tau}(ch) = CQ^{init}$, as a good channel.

To transfer channel qualities from end nodes to the coordinator, we can use an array of 16 bits in which each bit refers to the condition of the corresponding channel. This leads to 2 bytes communication overhead per packet to transmit channel conditions sensed by each node to the coordinator. This two bytes can be added to the payload of any layer in the protocol stack, from application to MAC. As we use one-hop communications to evaluate this technique, here we use two bytes of the MAC payload to attach this channel quality data to packets towards the coordinator. Note that this technique is only applicable for the nodes that have traffic towards the coordinator. Moreover, using MAC payload only works if all the data communications towards the coordinator are single-hop. Otherwise, for multi-hop networks, this data should be attached in the upper layers such as network layer. However, the exact attachment mechanism is not specified here and is an implementation decision. By each packet reception at the coordinator from node i, the CC_i report that is attached to the MAC payload is extracted and is added to a report list. Node ID i is also attached to the CC_i value in this list and there can be only one CC_i connected to a node ID. Thus, old CC_i values of a node will be replaced by new values. All the reports in the report list are periodically used to update the CQE of each channel and then the list will be cleared. Considering a slotframe as a period in which all the transmissions repeat in the network, we do this process at the beginning of each slotframe. At first we average the CC_i values of different nodes for each channel to have a global notion of channel conditions in the network area.

$$CC_{avg}(ch) = \frac{\sum_{i=1}^{n} CC_i(ch)}{n} \tag{4.12}$$

Here n refers to the number of CC_i reports in the report list that is the number of nodes that sent a channel condition report to the coordinator during the last slotframe period. After computing the CC_{avrg} of each channel, we map it into the range of EDs and apply it to the CQE of that channel (that is continuously updated by NICE), using a predefined weight γ and its previous value as follows:

$$CQE_\tau(ch) = \gamma\ CC_{avrg}(ch)\,(ED^{Max} - ED^{Min}) + (1-\gamma)\ CQE_{\tau-1}(ch) \tag{4.13}$$

This CQE_τ value is used as the input of the whitelisting algorithm. Accordingly, DCS results are combined with NICE results. Thus, when a channel is detected as a bad

quality channel (due to results of DCS technique) and is blacklisted, NICE still measures its quality and updates the CQE assigned to it. Therefore, after some time the previously blacklisted channels get a chance to again be used for communications to check if the observed interference by DCS is gone. This only happens if NICE realizes that the quality of a blacklisted channel is better than a channel under use. Re-introducing channels to the whitelist is crucial for the networks with dynamic interference conditions. In the following we discuss about the central whitelisting technique that is used at the coordinator node.

4.3.4 Channel Whitelisting

Whitelisting is performed periodically by the coordinator of the network to select a subset of good quality channels as the HSL for the TSCH protocol (lines 21 to 26 of Algorithm 1). The result of NICE and DCS, which are combined as a unique CQE array, is used as the input of this algorithm.

There are different approaches to do whitelisting, from threshold-based to ranking-based. To select a proper technique, a few constraints should be taken into account to do whitelisting for a TSCH network. First, if the whitelist size (|HSL|) is not prime to the slotframe size (L_{SF}), each timeslot only touches a subset of the channels in the HSL, not all of them. This causes a non-uniform chance of failures for different timeslots over time, because they may use different subsets of channels with different qualities. Furthermore, if an allocated link to a timeslot experiences persistent multipath fading on a channel, due to the use of a small subset of channels for communications of that link, its packet error rate will be increased. Here we use a fixed size whitelist, but based on the user requirements it is also possible to use a variable size whitelist with sizes that are prime to the slotframe size.

It should be considered that smaller whitelist sizes reduce the maximum number of channel offsets that can be used in the schedule. If more than this maximum channel offsets are used, two different links assigned to different channel offsets but same timeslot offset use the same channel for communications. This causes internal interference and packet losses. Accordingly, smaller whitelist sizes reduce the number of parallel communications that can be established at one timeslot, and also the overall throughput of a TSCH network. In threshold-based approaches, the number of channels with a better quality than a specified threshold may be very low. Thus it may not provide enough channels to meet a specific whitelist size. On the other hand, by using the NICE and DCS techniques, we assigned qualities to all of the channels and we can use these values to sort them. Therefore, we use a ranking technique to select a fixed number of best quality channels as the HSL for ETSCH. As IEEE 802.15.4 [7] standard suggests to use a pseudo-randomly shuffled set of all of the available channels for HSL, techniques such as the one presented by Shih et al. [66] can be used to shuffle the list of channels without regeneration overhead. The resulting shuffled HSL is used by the TSCH protocol for the hopping procedure.

4.3.5 EB Whitelisting

The coordinator device of the ETSCH network periodically uses whitelisting to extract the best HSL. The HSL and other information of the network such as link allocations and ASN are disseminated via the EBs defined in the TSCH protocol. We set up the coordinator to broadcast EBs periodically in the first timeslot of the slotframe with highest priority. Periodic transmission of EBs helps all devices in the network to synchronize with their coordinator at the start of each slotframe and to be aware of changes in the network setups.

When a coordinator broadcasts an EB with an updated HSL, there is a possibility of missing this EB in one or more devices. Using unicast and ACK-enabled communications for transmitting EBs, comes at a throughput cost to the network. The work in [85] shows that some of the IEEE 802.15.4 channels are affected less than the others by coexisting Wi-Fi networks which are the main source of interference on IEEE 802.15.4 channels (i.e., channels 11, 15, 20, and 26). Thus we decide to use a second, less dynamic, hopping sequence list consisting of a small subset of best quality channels to disseminate EBs in ETSCH. The EBSL is defined by the coordinator and has a fixed size of 4. This size is in line with the fact that four 802.15.4 channels typically are less affected by coexisting Wi-Fi networks, compared to others [85]. We do not limit the EBSL to only those four channels though, to ensure that always the best channels are selected. Therefore, the operating channel to transmit an EB for a given ASN and size of the slotframe (L_{SF}) can be computed as:

$$Channel(EB) = EBSL[\lfloor ASN/L_{SF} \rfloor \ \% \ 4] \tag{4.14}$$

We update this EBSL in a one-channel-per-period manner every time the main HSL is updated. In this method, every time the coordinator wants to broadcast an EB containing an update of the main HSL, it only updates the EBSL entry which was used for the last EB transmission. The process of updating this list is described in lines 27 to 35 of Algorithm 1. This algorithm finds the channel with the best quality, which is not in the EBSL, then puts this channel in the last used entry of the EBSL. This updating method reduces the possibility of burst EB losses in a joined device by only using best quality channels. Hence, when a device misses an EB which contains an updated HSL, it has a high chance to receive it in the later slotframes and synchronize its HSL to the network.

Timeslot phase difference caused by clock drift between a device and the coordinator can lead to disconnection of the link between them. This leads to burst EB losses even when the EBSL is the same at both. To solve this problem, we take channel 26, which is a non-overlapping channel with Wi-Fi, as a permanent member of the EBSL. This channel is considered to be the least noisy channel in urban environments. Every time a joined device experiences a burst EB loss equal to a predefined number N_{BL}, it considers this situation as a synchronization loss caused by timeslot phase difference and starts a passive scan on channel 26 to be synchronized again with its coordinator. Because the size of the EBSL is 4, it is possible for a device to receive an EB on channel 26 after a maximum time of 4 beacon periods, assuming no packet losses. If packet failures happen on EB reception on channel 26, the joining device should wait for another 4 bacon periods to again have the chance to receive an EB and join the network.

Table 4.2: List of Acronyms

Acronym	Description
ETSCH	Enhanced Time-Slotted Channel Hopping
NICE	Non-Intrusive Channel-quality Estimation
HSL	Hopping Sequence List
EBSL	Enhanced Beacon hopping Sequence List
DCS	Distributed Channel Sensing
ETSCH+DCS	ETSCH plus DCS
ETSCH–EBSL	ETSCH without EBSL
ED	Energy Detection
NG	Noise Generator
PRP	Packet Reception Probability which is extracted by simulations
PRR	Packet Reception Ratio which is extracted by experiments

4.4 Performance Evaluation

We investigated the performance of ETSCH and DCS through various experiments and simulations. For the experiments we use Atmel kits [50] introduced in Chapter 2. We also exploit some controlled noise generators to mimic the out-of-car interference in automotive WSNs. In reality, an IEEE 802.15.4 network may observe interference on multiple adjacent channels from a coexisting Wi-Fi device. Accordingly, each noise generator provides controlled interference by transmitting dummy packets on a pair of adjacent IEEE 802.15.4 channels. To implement this mechanism, a noise generator transmits a short packet on a channel and immediately hops to the paired adjacent channel. This process is done continuously to generate interference on both the paired adjacent channels only by one noise generator. Furthermore, each noise generator is programmed to hop to different pairs of adjacent channels within predefined periods to mimic the out-of-car interference dynamism.

The DCS technique is proposed for situations where there is an interference source around the network that is hidden from the coordinator. Actually, when there is no hidden interference source, the DCS technique has no effect on the channel whitelist, and whitelisting only follows the output of NICE. Based on this, we define two evaluation sets to study the performance of 1) the ETSCH technique without any interference hidden from the coordinator and 2) the ETSCH+DCS technique under existence of hidden noise. In the first evaluation set, we compare the performance of ETSCH with ATSCH, and basic TSCH using experiments and simulations. For the second evaluation set, we use experiments and focus on the performance evaluation of the DCS technique, in presence of hidden interference which is not detectable by NICE. Because DCS is an additional technique on top of ETSCH, we also perform the same experiments for ETSCH as well as TSCH to compare the results.

We introduced a number of acronyms during description of the proposed techniques in this chapter. To ease the understanding of the analysis, Table 4.2 gives a list of used acronyms in this chapter and their descriptions.

4.4.1 ETSCH Performance Evaluation

This subsection evaluates the performance of ETSCH (skipping the DCS technique) in comparison with other channel quality estimation technique called ATSCH and also the TSCH protocol. Since co-channel wireless interference is the main source of packet errors in urban networks, whitelisting can reduce this negative effect by using good quality channels. The level and also dynamism of interference can affect the performance of the whitelisting technique. To test the performance of ETSCH, we use experiments with controlled noise generators as well as simulations with real-world in-vehicle interference on IEEE 802.15.4 channels. We try different interference scenarios with different levels and dynamism of interference for the network under test.

Lab Experiments, Setup and Analysis

For the experimental evaluation of ETSCH, we use a mesh TSCH network with seven devices and one PAN coordinator. Motes are distributed in random places in a 10 $m \times$ 10 m office workspace. We use a wider area for our experiments, compared to an in-vehicle network in a truck. This is because due to obstacles (such as body of the car and passengers) in an automotive testbed, quality of links are usually lower than an office workspace. Instead of obstacles, we use longer distances between motes to reduce the link qualities in the experiments. We ran the experiments using a complete ETSCH with NICE and EBSL, and also a reduced version of it without the EBSL, called ETSCH–EBSL (ETSCH minus EBSL) in this section for ease of reference. ETSCH–EBSL uses the basic HSL with all 16 channels to transmit EBs. This allows us to investigate the impact of the EBSL on the performance of the network. We also implemented ATSCH [23] on top of our TSCH implementation to use it for our performance evaluations.

Slotframes of size $L_{SF} = 11$ are used in the experiments to be prime to the hopping sequence list size of 8. The first timeslot is allocated to EB transmission by the coordinator; 7 other timeslots are allocated to the end devices to transmit their packets of size 100 bytes. The three last timeslots are idle. Since ATSCH needs two more timeslots per slotframe to perform EDs, we use two of the three idle timeslots for it. Each experiment lasts for 6000 slotframes, thus each mote broadcasts 6000 packets in an experiment. All motes listen to all the timeslots for packet reception from other motes. Doing this, we can extract the quality of all available links in the network, as we build a full mesh network.

To simplify multiplications in all the used exponential smoothing equations, we use bitwise shift by using values of a power of 2 for all the used coefficients. This minimizes the processing overhead on the sensor nodes. We run a number of experiments with different α values to find a proper value. A value of α around 0.1 was found to have the best results. Thus, $\alpha = 2^{-3} = 0.125$ is used for experiments. The value of other parameters used in this experiment is shown in Table 4.3.

We consider four interference scenarios in our experiments; high, medium, low, and no interference. In the no interference scenario, we run the experiments without any controlled noise generator to see the cost of periodic HSL changes on the performance of our mechanism. Table 4.4 provides a short description of each scenario.

As shown in Chapter 3, a WSN in a moving car constantly experiences interference

Table 4.3: Setup for basic ETSCH evaluation

Parameter	Value	Description
α	1/8	Exponential smoothing exponent for NICE
Tx power	0 dBm	Transmission power of sensor nodes
Packet size	100 Bytes	Size of the data packets at MAC layer
$\|HSL\|$	8	Hopping sequence list size
$\|EBSL\|$	4	Enhanced beacon sequence list size
L_{SF}	11	slotframe size (number of timeslots per slotframe)
Whitelisting period	10	Whitelisting period in terms of slotframe duration
N_{BL}	5	Number of burst EB losses to do re-synchronization

Table 4.4: Interference Scenarios

Scenario	Noise behavoir
No interference	no controlled NG
Low interference	noise on 2 channels (1 NG), hop every 20 seconds to new channels
Medium interference	noise on 6 channels (3 NGs), hop every 20 seconds to new channels
High interference	noise on 6 channels (3 NGs), hop every 5 seconds to new channels

from different sources (e.g., Wi-Fi networks). Assuming that each interference source is visible over a range of up to 50 meters, and this car moves with a speed of 36 km per hour, each noise source would be visible for 5 seconds. Our high interference scenario models this kind of interference when there are three interference sources visible at any time which each generate interference on two IEEE 802.15.4 channels. For the medium interference scenario, we consider lower mobility of vehicles which leads to increasing the visibility duration of each interference source. Thus, we have lower dynamism of interference in this scenario. In the following, we analyze the results of this experiment set for different interference scenarios.

Figure 4.5 shows the average of achieved PRR of all links in the network for different mechanisms and interference scenarios. Both versions of ETSCH provide better PRR on average in comparison to TSCH and ATSCH, when the network experiences dynamic interference. This shows the effect of highly adaptive channel quality estimation that is realized by NICE, which selects the best quality channels for hopping. As depicted in Figure 4.5, ATSCH performs almost the same as basic TSCH. There are two reasons for such a result: 1) The rate of channel samplings by ATSCH is much lower than for ETSCH. Therefore, it can only deal with very low interference dynamics, and cannot detect and follow the highly dynamic interference (which exists in in-vehicle networks). This leads to increasing packet losses when noisy channels are also selected to be used in the HSL. 2) ATSCH does all the samplings in one timeslot every slotframe. Our NICE technique spreads channel samplings over a slotframe and therefore it can detect noisy channels better.

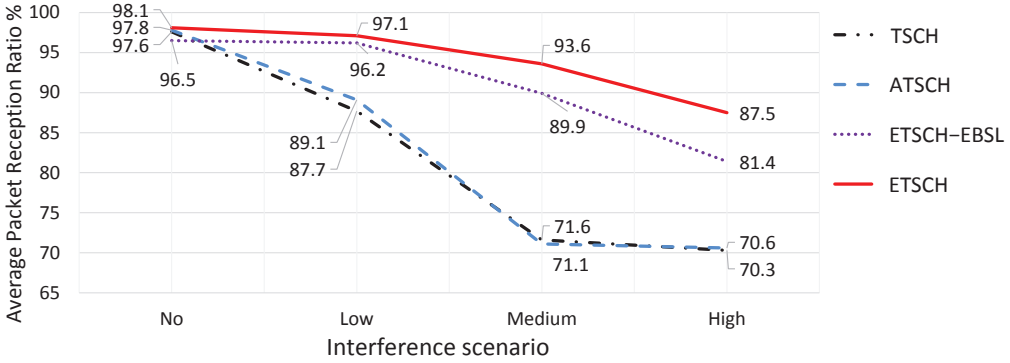


Figure 4.5: Average achieved PRR of different mechanisms for different interference scenarios

Figure 4.5 also shows that using an EBSL to disseminate EBs improves the PRR of ETSCH compared to ETSCH–EBSL in all the interference scenarios. This is because it reduces the possibility of EB losses, and accordingly HSL mismatches between the coordinator and nodes.

To better investigate the behavior of different mechanisms, Figure 4.6 shows the distribution of average PRR of all links in the network over a window of 500 transmissions, for the different mechanisms and scenarios. The results show that using an EBSL generally leads to a lower standard deviation in the PRR results. This guarantees a higher reliability level for the links. Figure 4.6(a) shows that for the scenario with no controlled noise generator, the standard deviation of ETSCH–EBSL results is higher than that for TSCH. This is because TSCH always uses the same HSL for hopping, while ETSCH–EBSL may use different sets of channels as the HSL, due to detecting low noise on some channels. In the case of missing one EB by a node, which can be due to transient interference or multipath fading, there is a possibility of HSL mismatch between that node and other nodes in the network. This situation leads to packet drops on incoming and outgoing links of this node, which is due to using different channels by the source and destination on the link. It continues until the node detects this mismatch and synchronizes its HSL with that of the coordinator. This synchronization may take a few slotframes and reduces the PRR of those links for that time window. By using EBSL in ETSCH, after missing one EB by a node, it still has the chance to receive the EB on the next 3 slotframes and synchronize its HSL with the coordinator. This reduces continuous packet losses on each link and thus provides a higher PRR as well as a lower standard deviation in the results of ETSCH for scenarios with interference.

Figure 4.7 illustrates the distribution of maximum length of burst packet losses over a window of 500 transmissions, considering all links of the network in each mechanism. This shows the continuity of correct service of all links in the network, in which any link may be a candidate to be used for dedicated communication in the real-world networks. Furthermore, we extract the maximum length of burst packet losses in windows if 500 transmissions. Thus, for each link we have 12 values of maximum length of burst packet losses (6000 packet transmissions for each node). This shows the occurrence of communication problems during time on each link and helps to better perceive the

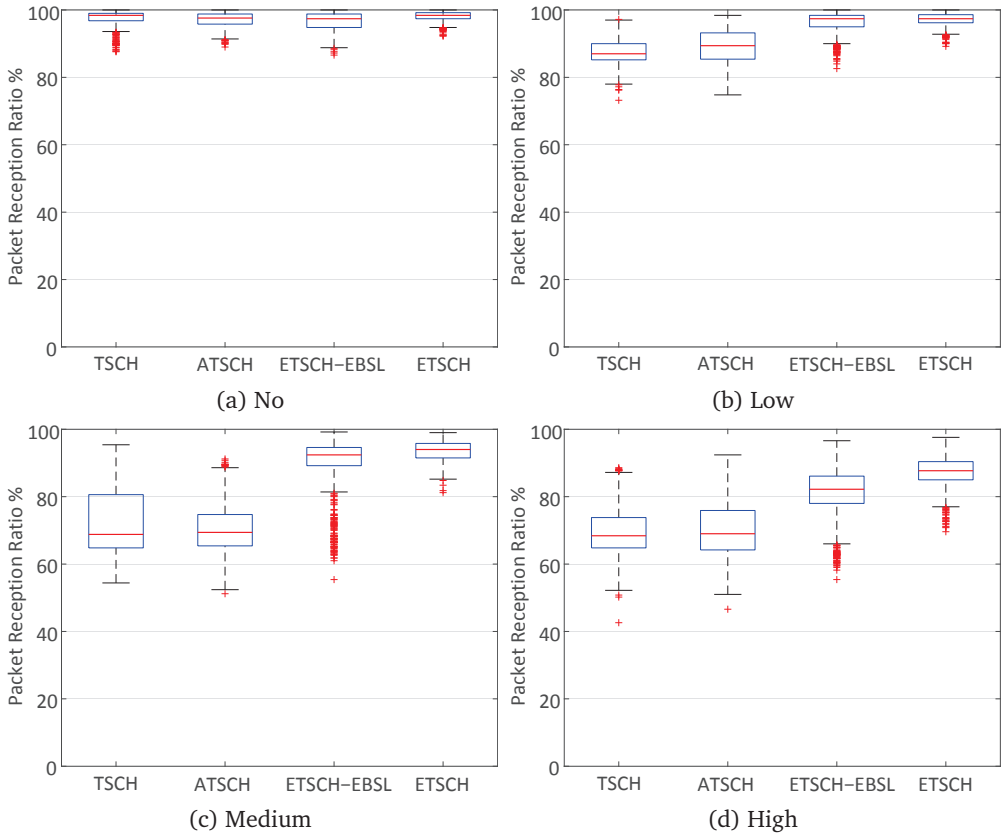


Figure 4.6: PRR distribution of all network links, over a window of 500 transmissions, for different mechanisms and different interference scenarios.

performance of communications.

Figure 4.7(a) shows the measured results for the no interference scenario. Compared to basic TSCH and ATSCH, for both versions of ETSCH there is a slight increase in the maximum length of the burst packet losses in absence of interference. Due to the higher number of channel samplings by ETSCH, it experiences fast changes of the assigned quality to channels caused by small quality variations. This leads to more frequent HSL changes for ETSCH that lead to a higher chance of HSL mismatch between each device and coordinator. By considering the outliers in this scenario, which are the bottlenecks for maximum length of burst packet losses, ETSCH still outperforms TSCH and ATSCH.

For the second interference scenario (Figure 4.7(b)), plain TSCH has maximum burst packet losses with a median length of two. This is because TSCH hops over all channels and there are only two adjacent channels with interference which cause packet losses. ATSCH has no gain compared to the results of TSCH. The reason is the slow channel quality estimation process, which causes some delay in detecting noisy channels and some packets to be lost. Both versions of ETSCH follow the dynamism of interference

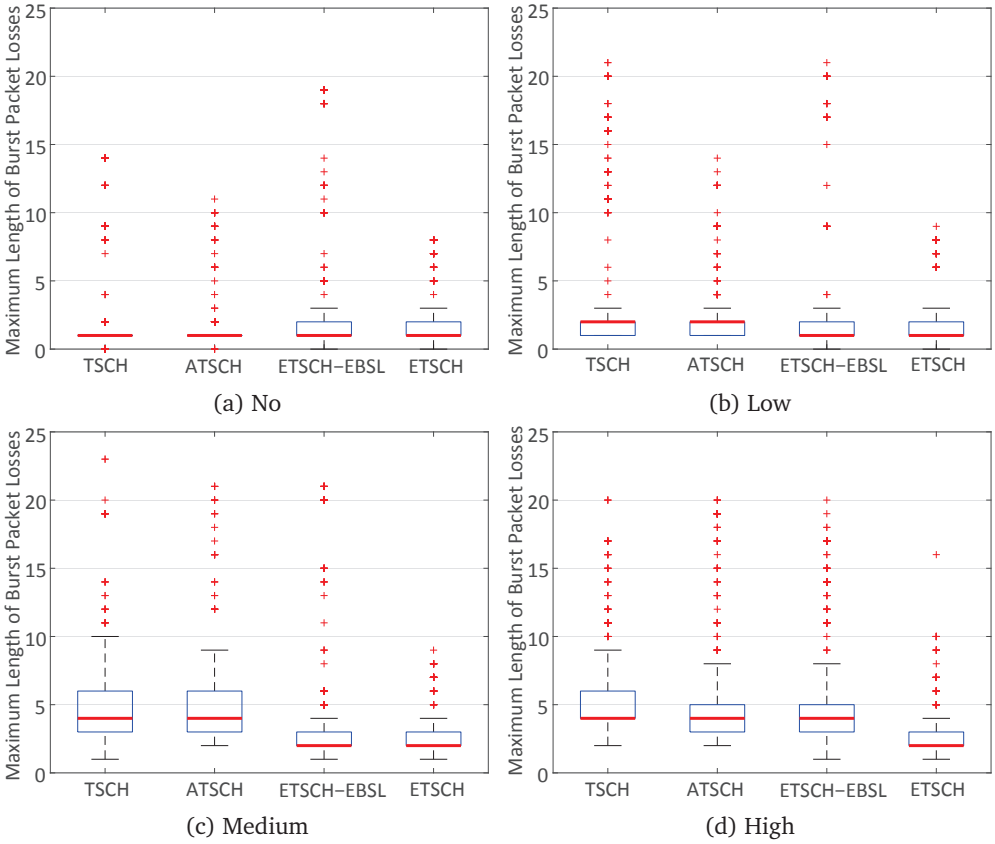


Figure 4.7: Maximum length of burst packet losses of all links over a window of 500 transmissions, for different mechanisms and different interference scenarios.

faster, and thus result in smaller median value of burst packet losses. Considering the outliers, ETSCH–EBSL experiences occasional burst packet losses with higher length, compared to ATSCH. This is due to more frequent HSL changes that cause HSL mismatches between devices and the coordinator when there is an EB loss. However, ETSCH can reduce this kind of burst packet losses by using the EBSL, synchronizing the HSL in a more reliable way.

Figure 4.7(c) shows that when the number of noise generators increases to three (6 noisy channels), TSCH and ATSCH experience higher median value and a wider distribution of the maximum length of burst packet losses. For TSCH which always uses the same HSL for hopping, having more noisy channels increases the possibility of using consecutive noisy channels for transmissions on a link. For ATSCH which has a slow interference detection process, using noisy channels in the HSL and HSL mismatch between nodes and the coordinator, due to EB losses, are causing bursts. ETSCH–EBSL decreases burst packet losses by performing more channel samplings and thus detecting noisy channels faster than ATSCH. ETSCH has even lower maximum length of burst

packet losses with less deviation, because it reduces EB losses which are the source of HSL mismatch between nodes.

For the high interference scenario (Figure 4.7(d)), all techniques except ETSCH perform almost the same as plain TSCH. This is because of highly dynamic interference which causes EB losses and problems for HSL synchronization between coordinator and devices. Even in this scenario, ETSCH decreases the maximum length of burst packet losses compared to other mechanisms (for median value, normal distribution, and outliers). This is done by keeping the network nodes synchronized, using best channels to transmit EBs.

Simulations with Real-World Interference Data

We implemented the functionality of ETSCH and ATSCH on top of the Matlab TSCH simulation framework that is presented in Chapter 3. We use the reported data sets of different scenarios to evaluate the performance of our technique under real-world interferences. Because these data sets show the interference at one point in a car, simulations are only valid for the communications towards one node. Therefore, we consider a star TSCH network with a number of nodes and one PAN coordinator, where each node has a link towards the coordinator. We consider the same distance of 3 meters between all the devices and the coordinator, path-loss exponent of 3.5, and transmission power of 0 *dBm*.

As for the lab experiments, slotframes of size $L_{SF} = 11$ are used in the simulations. The first timeslot is assumed to be used for EB transmission by the coordinator to broadcast HSL (and EBSL). The other timeslots in each slotframe are dedicated for transmission of packets (with length 100 bytes) by the network nodes (10 nodes in TSCH and ETSCH simulations and 8 nodes for ATSCH simulation) to the coordinator. In ATSCH, the last two timeslots of each slotframe are used for performing EDs. Based on the length of available data sets, each simulation lasts for 300 seconds. Thus, there are 30000 timeslots in one simulation. Other parameters are configured as in the previous experiments (Table 4.3).

The channel offset, which is used for a TSCH slotframe, can affect the communication performance. This is due to the possible periodic behavior of a noise signal on a channel. Because a network can define multiple slotframes with different channel offsets, we consider all possible parallel transmissions on different available channels to compute the PRP of each timeslot. Accordingly, for each timeslot we compute the PRP of transmissions on all available channels in the HSL and then use the average of them as PRP of that timeslot.

Figure 4.8 shows the simulation results for different mechanisms and different interference scenarios. Five interference scenarios are considered; i.e., Wi-Fi and Bluetooth file transfer between two devices in a car, driving along a road near some apartments and an office area downtown, and a lifelike scenario which is a mix of the Wi-Fi, Bluetooth, and downtown scenarios. As the results show, ETSCH outperforms ATSCH and TSCH in term of PRP, in all scenarios. For a scenario like Bluetooth, where interference affects almost all the channels, using a whitelist is not much effective. This is because interference is distributed almost uniformly over all channels and whitelisted channels perform almost the same as blacklisted channels. For other scenarios, in which

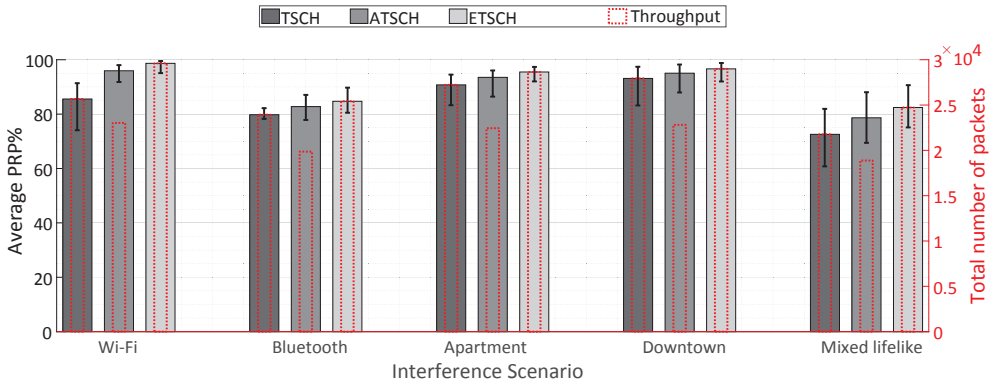
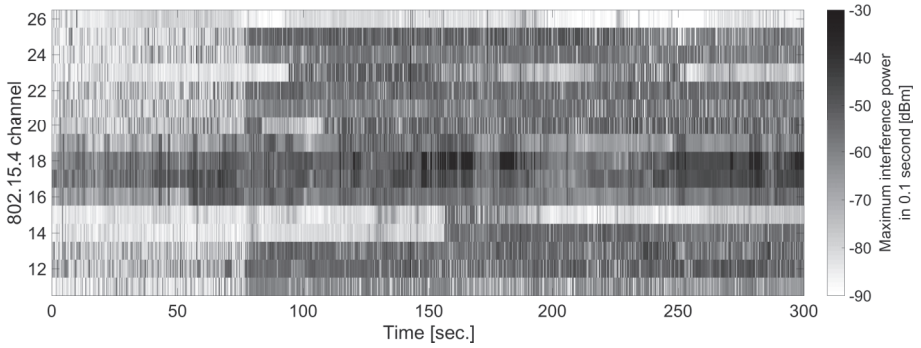


Figure 4.8: Average and distribution of PRP over the 300 seconds of simulation together with achieved throughput in term of the total number of timeslots with successful packet transmissions, for different mechanisms and different interference scenarios.

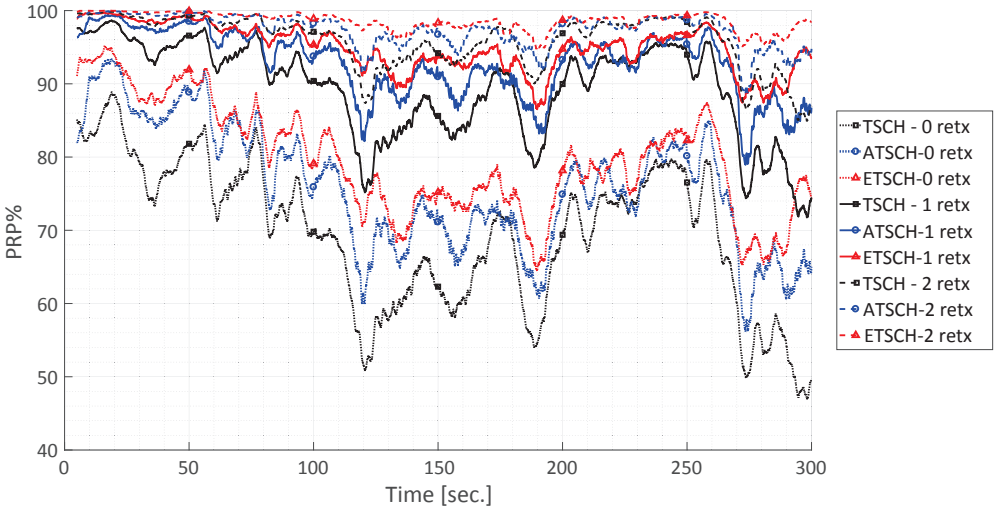
the distribution of interference over channels is not uniform, both ATSCH and ETSCH techniques perform better than the plain TSCH. While ETSCH outperforms ATSCH in terms of the average PRP in these scenarios, it also has a lower standard deviation, which shows its continuity of correct service over time. For the Bluetooth scenario, both ETSCH and ATSCH have a higher standard deviation of PRP compared to TSCH. This is because Bluetooth also uses a channel hopping scheme, together with an adaptive channel whitelisting. This causes the coexisting ETSCH network to perform good when whitelists of Bluetooth and ETSCH are not overlapping, and to perform bad when whitelists are overlapping.

Figure 4.8 also depicts the total number of timeslots with successful packet transmissions to give an estimation of the achievable throughput. This value is calculated based on the number of available timeslots for communications and the computed average PRP for each scenario. ATSCH has a lower throughput than the plain TSCH in all scenarios. This is due to the overhead of idle timeslots that are used by ATSCH for channel estimation. On the other hand, ETSCH increases the throughput of the network by increasing the PRP, while NICE has no throughput overhead.

To study the PRP over time, Figure 4.9 shows the interference impact on the PRP of different mechanisms with zero to three retransmissions, for the lifelike (mixed) scenario presented in Chapter 3. This figure confirms the results of Figure 4.8 and also shows that, depending on the interference conditions, PRP may changes a lot over time. As Figure 4.9(b) shows, ETSCH detects noisy channels at the beginning of the simulation very fast (about 1 to 2 seconds), while it takes about 10 seconds for ATSCH to detect the noisy channels and follow interference. Compared to the plain TSCH, the whitelisting techniques used by ATSCH and ETSCH may provide a considerable increase in average PRP in some periods of time (e.g., time 100 sec. to 160 sec.). This increase in PRP may also be very low in some periods (e.g., time 200 sec. to 250 sec.). This behavior highly depends on the number of noisy channels at each point of time, which makes whitelisting very effective when the distribution of noise over different channels is not



(a) Interference behavior over 300 s using HeatMap plot.



(b) Moving average of PRP over a window of 500 timeslots, for different mechanisms and number of retransmissions.

Figure 4.9: Effect of the mixed lifelike interference on the IEEE 802.15.4 channels and performance of different mechanisms under this interference, extracted by simulations.

uniform. It is also possible that the used whitelisting technique performs worse than the basic TSCH for short periods of time (e.g., ATSCH at time 230 sec.). This happens when the set of noisy channels changes quickly and overlaps with the selected whitelist. Thus, there will be packet errors until the new noisy channels are detected and a new whitelist is picked.

Figure 4.9(b) also shows the effect of retransmissions on the reliability of communications. To simulate $N_{ReTx} > 0$ retransmissions of a packet, we consider every $N_{ReTx} + 1$ consecutive timeslots (starting from timeslot τ) for transmission of one packet and cal-

culate the PRP of those timeslots using Equation (4.15).

$$PRP^{\tau}_{ReTx} = PRP^{\tau} + \sum_{k=1}^{N_{ReTx}} \left(PRP^{\tau+k} \times \prod_{i=0}^{k-1} (1 - PRP^{\tau+i}) \right) \tag{4.15}$$

As it is shown, retransmissions considerably improve the PRP of the communications for all techniques. While employing more retransmissions improves the reliability of communications, it also increases the latency of communications. This is because a packet should be buffered until the next assigned timeslot to it to be retransmitted. Thus, according to the latency requirements of the application and also the TSCH schedule, the number of allowed retransmissions is usually limited. Figure 4.9(b) shows that ETSCH with one retransmission has very close PRP to the basic TSCH with two retransmissions. This shows the positive effect of the ETSCH technique on both reliability and latency of the TSCH protocol. Moreover, ETSCH with two retransmissions provides average PRP greater than 95% all the time, while basic TSCH and ATSCH still experience fluctuations of 10% sometimes.

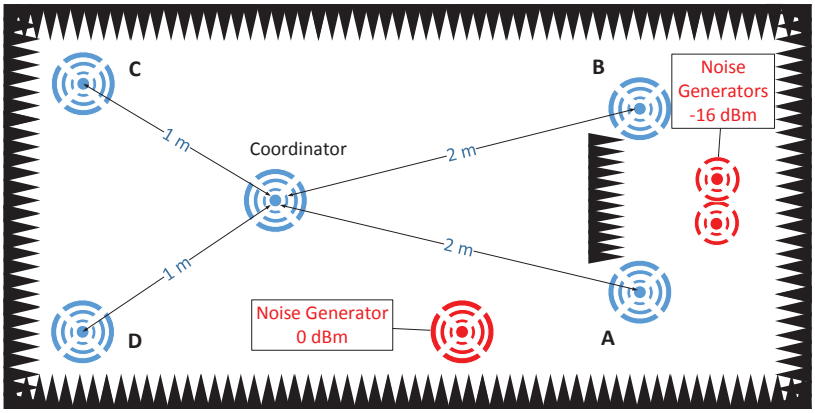
We conclude that real-world interference has a high impact on the performance of TSCH protocol. Techniques such as ETSCH can improve the reliability of this protocol under real-world dynamic interference. However, employing retransmission is necessary to provide higher reliability levels for the network.

4.4.2 ETSCH+DCS Performance Evaluation

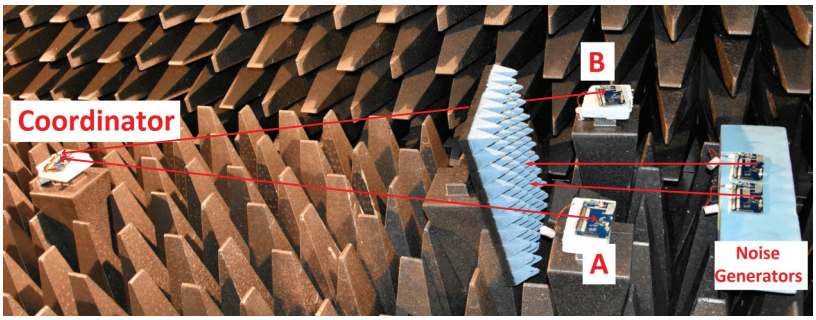
In this section we evaluate the performance of ETSCH when DCS is enabled, in presence of hidden interference from the coordinator. The interference data sets that are introduced in Chpater 3 are only for a single point in a car. However, for our evaluations in this section, we need interference at multiple points to simulate the hidden interference problem. Therefore, we conduct experiments for DCS performance evaluations, and skip simulations in this section.

Evaluation Setup

We use a mesh TSCH network with four devices and one PAN coordinator for this set of experimental evaluation. Motes are deployed in an anechoic chamber as depicted in Figure 4.10(a). These experiments are performed in an anechoic chamber to implement a scenario with hidden interference. To do this, we used two noise generators with a low transmission power of -16 dBm. As depicted in Figure 4.10(a), these noise generators are placed close to each other at a distance of 0.5 m of two end nodes. A signal absorber wall is placed between the coordinator and these noise generators to prevent a line of sight radiation path between them. Because there is no reflection in the anechoic chamber, the interference is completely hidden from the coordinator. This placement causes a level of interference for those two nodes that are close to the noise generators, while this interference at the coordinator would be almost invisible. Figure 4.10(b) shows a part of the actual setup, including the coordinator, nodes A and B, and two noise generators. Using this scenario, we can study the performance of the DCS technique, while NICE cannot detect this interference at the coordinator point. To study the performance



(a) Node deployment map



(b) Picture taken from part of the actual setup

Figure 4.10: Node deployment to evaluates the performance of DCS technique on top of ETSCH.

of NICE when DCS is on, we use one noise generator at a transmit power of 0 dBm, which is visible to all of the network nodes. All of these noise generators use a period of 20 seconds to hop to new channels, in order to generate dynamic interference.

We ran the experiments for ETSCH with and without DCS. This allows us to investigate the impact of the DCS technique on the performance channel quality estimation while there is noise hidden from the coordinator in the network. To have a complete performance study, we also ran the experiments using the basic TSCH protocol. As in the first experiment set, each experiment lasts for 6000 slotframes, thus each node broadcasts 6000 packets in an experiment. Each node also listens to all other timeslots for packet reception from other nodes.

Parameter Setup

Table 4.5 shows the used values for all the parameters in this scenario. Like what we did for choosing the value of α in Table 4.3, we run a number of experiments with different combinations of α' and β' values to find a suitable value for them. Because the optimum values for these parameters depend on the dynamism level of the interference,

Table 4.5: Setup for ETSCH evaluation

Parameter	Value	Description
α'	1/8	Exponent used for increasing CQ
β'	1/4	Exponent used for decreasing CQ
γ	1/8	Weight used to apply CC_{avrg} into CQE
θ	128	CQ threshold used for DCS
CQ^{init}	180	Initial value of CQ for DCS
CQ^{Max}	255	Maximum value of CQ for DCS
CQ^{Min}	0	Minimum value of CQ for DCS
L_{SF}	5	slotframe size (number of timeslots per slotframe)

we tried different interference scenarios and picked the values which have the best results for all scenarios. As it is shown in the table, we pick two different exponents for CQ calculations at each node for the DCS mechanism (α' and β'). The exponent that is used for positive samples (α') has a lower value than the exponent used for negative samples (β'). This is because we want to make negative or channel_busy samples more effective in CQ calculation to detect the noisy channels faster. It prevents long burst transmission failures on a noisy channel by reducing its CQ below the threshold θ after a few subsequent negative channel assessments, and report it to the coordinator as a noisy channel. We consider the value of threshold θ to be the mean of CQ^{Min} and CQ^{Max}. By defining the CQ^{init} to be greater than θ, an initial chance is given to a newly used channel to be considered as a good channel, even if a few transmission failures happen. This is because a newly added channel to the HSL is assumed to be a good (low-noise) channel.

The coordinator determines the quality of each frequency channel (CQE) based on the output of NICE as well as the result of the DCS technique. As stated in Equation (4.13), a weighted average is applied with γ as the weight for the DCS result. Considering that the results of DCS are mapped to the range of ED, and observing that in our ETSCH experiments the CQE value of each channel was normally in the lowest 1/8 part of the ED range, we pick value of 1/8 for γ. Using this value, the effect of NICE and the DCS technique on the computed CQE of each channel is expected to be almost the same. Other parameters are the same as for the first experiment set (Table 4.3).

Performance Analysis

We investigate PRR to evaluate the performance of the DCS technique. Figure 4.11 shows the distribution of PRR of all links in the network for different mechanisms. We included one experiment with the TSCH protocol without any Noise Generator (NG). This experiment is done to check if the deployed network works fine and the placement of nodes does not negatively affect the performance of communications between them. A window of 500 transmissions is used to plot the results. As Figure 4.11 shows, when there is no noise generator in the network area (TSCH-No NG), the median value for PRR is 100% and all links in the network are perfect. As depicted in this figure, on average ETSCH provides better PRR in comparison with TSCH. This is because ETSCH can

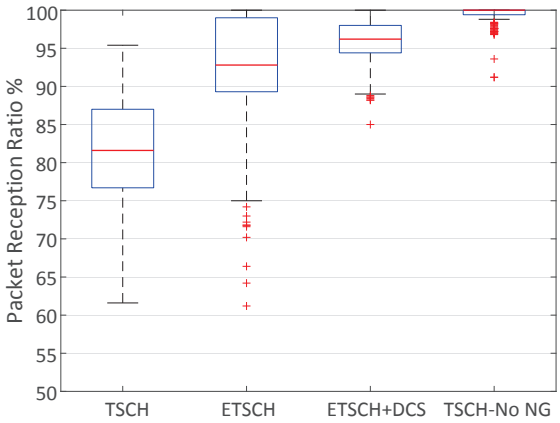
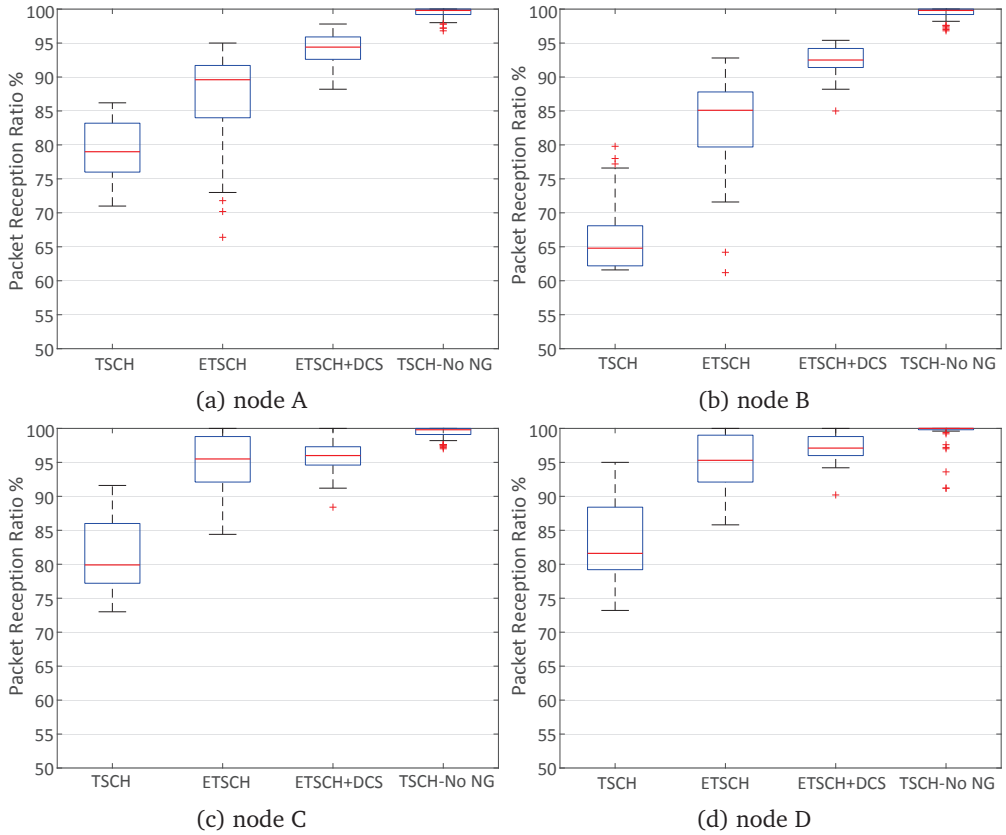


Figure 4.11: PRR distribution (of all links in the network) over a window of 500 transmissions for different mechanisms in a setup with hidden interference; TSCH without interference (TSCH-No NG) is given as reference.

detect the visible noise generator and skip using those channels that are noisy because of it (as confirmed in the ETSCH evaluation of the previous section). The high variation in the results of TSCH and ETSCH is due to the different interference conditions at the point of different nodes. For example, nodes A and B experience interference from all three noise generators, while nodes C and D only are affected by one noise generator. This leads to a considerable variation in the quality of incoming links to each node and thus different PRR values for them.

The next observation of Figure 4.11 is that on average, ETSCH+DS provides higher PRR than ETSCH. This is because the noise that is generated by the two hidden noise generators on four random channels, only affects the incoming links of nodes A and B which leads to higher packet errors for these links. The DCS technique reports these noisy channels to the coordinator to be skipped for the channel hopping process. This leads to less packet errors and thus higher PRR for the mentioned links which leads to less variation in the results. Even by using the DCS technique, there is still some packet loss. These packet losses are due to the detection process of this technique which uses communication status to see if there is noise on a channel or not. This leads to some delay in detecting noisy channels.

To better investigate the behavior of different mechanisms, Figure 4.12 shows the PRR of all incoming links to each node in the network for different mechanisms. The figure shows that ETSCH provides better PRR than TSCH for all the nodes in the network. This PRR increase is less for nodes A and B compared to nodes C and D. This is because nodes A and B are affected by the two hidden noise generators and ETSCH cannot prevent their interference, while for nodes C and D, it is only the visible noise generator that affects their packet reception, which is handled by ETSCH. The difference between PRR of TSCH at nodes A and B, which have the same setup, might be due to the orientation of the noise generators' antennas which leads to different interference effects on each of the nodes. This is because of the non-perfect omni-directional antennas of the sensor nodes. The high variation in PRR of ETSCH at each node is due to



(a) node A (b) node B

(c) node C (d) node D

Figure 4.12: PRR distribution of incoming links to different nodes over windows of 500 transmissions for different mechanisms in hidden interference setup.

different distance between each couple of nodes. Because the transmission power of all the nodes is the same, different distances cause different signal to noise ratios at each receiver for different links. This leads to different PRR for incoming links to a node. For example, due to the short distance between A and B, there is a high chance that packets transmitted by B are correctly received at A, even in presence of interference. This is while for packets that are transmitted by node C, the greater distance leads to lower signal power at node A and thus lower signal to noise ratio and higher rate of packet errors.

In Figure 4.12, for ETSCH+DCS, the PRR of the links to nodes A and B is higher with smaller variation compared to ETSCH. This shows that the DCS technique detects the interference hidden from the coordinator at the point of other nodes. As mentioned, this technique cannot completely resolve the negative effect of those hidden interferers because it uses normal communications to detect if there is noise on a channel.

As Figure 4.12 depicts, ETSCH+DCS outperforms ETSCH also for incoming links to nodes C and D which are not affected by the hidden noise generators. The reason is the

CCA mechanism that is defined in the TSCH protocol. CCA is done before each packet transmission by each node, to skip transmissions if the used channel is busy. In this case, the outgoing links from both nodes A and B will skip transmission due to the busy_-channel result of CCA on the channels that are affected by the hidden noise generators. Thus, the PRR of links from nodes A and B to nodes C and D will be decreased for ETSCH. This is while the DCS technique reduces the CCA failures at nodes A and B and thus increases the number of received packets at nodes C and D. In conclusion, DCS is a useful technique to be used together with ETSCH to improve reliability of the network and reduce variation in PRR of different links.

4.5 Energy Consumption Analysis

Due to the heterogeneity in power sources of wireless nodes in automotive WSNs, energy may be an issue only for some nodes and others (including the coordinator which is connected to the central ECU of the vehicle) use a wire-based energy source from the vehicle. EDs that are performed by NICE and the two bytes field that is added to each packet for the DCS technique are the energy consumption overhead of ETSCH+DCS compared to the plain TSCH. Because NICE is used only by the coordinator of the network and energy may not be a stringent constraint for the coordinator, energy consumption is not a crucial metric in our work. However, to have a comprehensive comparison between ETSCH+DCS and other mechanisms, we analyze energy consumption.

The energy consumption for a given number of packet communications (E_{comm}) can be extracted by Equation (2.2). T_{listen} in that Equation is the duration that the receiver should listen to the medium to receive a packet which on average is

$$T_{listen} = T_{Tx} + (macTsTxOffset - macTsRxOffset) = T_{Tx} + 1.1ms \tag{4.16}$$

This is the packet transmission/reception duration plus the duration that a receiver wakes up before the transmitter starts packet transmission. Here, we ignore the switching time between different radio states. As the physical layer of the IEEE 802.15.4 standard defines, transmission of each byte takes 32 μs. Thus, transmission duration of a packet with length L_{packet} can be represented as

$$T_{Tx}[\mu s] = L_{packet}[Bytes] \times 32[\mu s/Bytes] \tag{4.17}$$

Since ETSCH+DCS and ATSCH perform a number of EDs per slotframe for channel quality estimations, they impose energy overhead as well. The energy consumption of EDs (E_{CQE}) can be calculated as

$$E_{CQE}[J] = \left(I_{ED} \times N_{ED} \times T_{ED}\right) \times V_{cc} \tag{4.18}$$

where I_{ED} stands for the radio transceiver current in energy detection mode and T_{ED} represents the duration of one energy detection. N_{ED} is the number of EDs performed during a given period.

The expected number of packet transmissions for a successful packet delivery over a link is equal to 1/PRP. We use this metric to consider the average energy consumption

Table 4.6: Performance overview of different mechanisms

Mechanism	PRP%	Expected Energy Consumption (mJ)	Overhead
TSCH	72.5	1.830	no
ATSCH	79	1.692	Two timeslots per slotframe
ETSCH+DCS	82.5	1.761	Two-bytes per packet

for packet (re)transmissions to deliver all packets to their destinations. Thus, the total energy consumption of each mechanism is

$$E_{mechanism} = \frac{1}{PRP} \times (E_{comm} + E_{CQE}) \tag{4.19}$$

Based on the ATmega256RFR2 datasheet, $I_{ED} = I_{Rx} = 12.5$ mA, $I_{Tx} = 10$ mA, $V_{cc} = 3.3$ V, and T_{ED} is 128 μs. Here, we use the PRP values extracted from simulations of the lifelike interference scenario (Figure 4.8). Considering that each slotframe consists of 8 timeslots in our experiments (skipping the three timeslots left idle to be used by ATSCH), the coordinator sends an EB with length $L_{packet} = 100$ bytes and receives 7 packets with the same size. With an average of 2.5 EDs per timeslot in ETSCH and PRP = 82.5%, the consumed energy by ETSCH+DCS in a slotframe is $E_{ETSCH+DCS} = 1.761\ mJ$ at the coordinator. TSCH does not perform any EDs and does not have the two bytes overhead of the DCS technique for each packet (packets with size of 98 bytes). Because it has a lower PRP of 72.5% compared to ETSCH+DCS, its energy consumption is $E_{TSCH} = 1.830\ mJ$ which is still more than the consumed energy by ETSCH+DCS. ATSCH uses two extra timeslots in each slotframe for EDs and has PRP of 79%. Thus, its energy consumption is $E_{ATSCH} = 1.692\ mJ$. This is 4% lower than the energy consumed by ETSCH+DCS in a slotframe, but in this example it comes with a bandwidth overhead of 20% for the network due to use of idle timeslots for channel quality estimation. Table 4.6 shows a performance overview of each mechanism.

Again we emphasize that the energy consumption overhead of the channel quality estimation is only for the coordinator node, which has less energy limitation compared to the other nodes in the network. Instead, higher PRP provided by ETSCH+DCS leads to lower number of required packet retransmissions by the end nodes. It means that the end nodes, which usually have stringent energy constraints, consume less energy when ETSCH+DCS is applied compared to TSCH and ATSCH.

4.6 Summary

This chapter proposes Enhanced Time-slotted Channel Hopping with Distributed Channel Sensing (ETSCH+DCS), a mechanism on top of the IEEE 802.15.4 [7] TSCH protocol that uses a combination of a central and a distributed channel-quality estimation technique. This mechanism extracts the quality of different wireless channels to select the channels with the lowest interference as the hopping sequence list to improve the performance of the TSCH protocol. The central channel measurement technique, called

NICE, operates at the coordinator of the network and proactively measures the spectrum energy in the idle part of each timeslot, when all the nodes in the network are silent. The energy sampling results are used to assign qualities to wireless channels. The distributed technique is performed by all the nodes in the network and uses the CCA results together with packet reception status to estimate the noise level on each channel. To use the packet reception status as a sign of interference on the communication channel, it is proposed to send a dummy packet with the shortest possible MAC header when there is no packet transmission request from the application layer. Based on the TSCH communication diagram, it is shown that transmitting a small packet consumes less overall energy than not sending a packet, because of reduced idle listening. The results of the centralized and distributed channel quality estimation techniques are used to assign a quality factor to each channel. Then, channels with better qualities are periodically selected as the Hopping Sequence List (HSL) of TSCH.

ETSCH+DCS also uses a small secondary hopping sequence list (EBSL), that consists of the best quality channels, to disseminate periodic Enhanced Beacons (EBs). These EBs contain control information of the network such as the HSL. Only one field of the EBSL is updated per period. The rate of EB losses in the network is reduced compared to using the regular HSL for broadcasting EBs. Experimental and simulation results show that ETSCH with NICE and EBSL provides higher packet reception ratios and lower length of burst packet losses, compared to the plain TSCH protocol and another related approach called ATSCH. Experiments also show that the DCS technique can detect existing interference in parts of the network that is not detectable by the centralized NICE technique, and thus it increases the PRR in those scenarios.

The evaluations show better performance of the techniques proposed in this chapter. However, simulation results based on the real-world data-sets (presented in Chapter 3) show that an ETSCH WSN may still face reliability issues during time. This makes it necessary to employ other techniques to improve the reliability of communications. One of these techniques is retransmissions at the MAC layer. It should be considered that each retransmission of a failed packet increases its delivery latency according to its buffering time. This may lead to invalidation of the data. Accordingly, the TSCH schedule should be managed carefully to reduce the communication latency. Lower communication latency enables more retransmissions to be used for delivering a packet, which increases the reliability of communications. Accordingly, in the next chapter, we introduce a topology management and TSCH scheduling technique to provide low latencies for TSCH communications in an automotive WSN.

5

Topology Management and TSCH Scheduling

5.1 Overview

In automotive WSNs the wireless sensor nodes usually are able to communicate directly with the Personal Area Network (PAN) coordinator. In such a platform, a star topology may be used to form the required network. Due to the high density of nodes in these WSNs, using contention-based MAC protocols may lead to considerable internal interference though. The TSCH protocol prevents such internal interference by using time-slotted communications with a predefined schedule. The scheduling task, which is the assignment of timeslots and channel offsets to the links of the network, is left to the upper layers in the protocol stack.

The communication schedule has a high impact on the performance of the WSN. There are several TSCH scheduling algorithms such as [8, 13, 25, 44, 55, 56] that aim at improving reliability and/or latency. These scheduling algorithms usually assume the network communication topology (which is determined by the network layer) as their input, and their output differs for different input topologies. When a WSN is small and dense, a wide range of topologies can be used for it. A suboptimal topology may lead to larger slotframes that causes higher end-to-end communication latency (especially when retransmissions are used), or unnecessary multi-hop communications that lead to higher latency and lower reliability. As an example (see Figure 5.1), using the TSCH MAC together with a star topology leads to a long slotframe and high packet delivery latency when the number of nodes is increased. On the other hand, a tree topology can reduce the slotframe size and accordingly the latency of communications by enabling parallel communications. This enables more retransmissions to be done within the required latency limits which leads to higher reliability of communications. Considering the reliability issue of low-power automotive wireless links due to external interference (as concluded in Chapters 3 and 4), retransmissions are necessary. This makes topology management and link scheduling necessary for automotive WSNs.

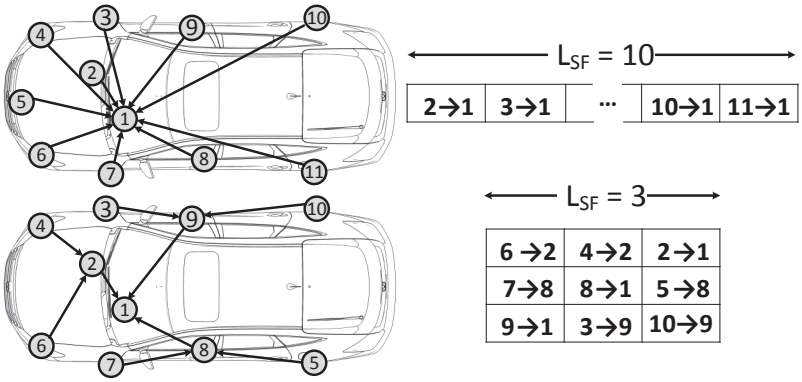


Figure 5.1: TSCH schedule for a star topology compared to a tree topology (L_{SF} is the slotframe size).

In an automotive WSN where all data is sourced or ends up at the main Electronic Control Unit (ECU), data convergecast is much more problematic than data distribution from the central node to the end-nodes. This is because for data convergecast, multiple nodes should send different data packets to a common node (sink node). This requires transmissions in a sequence to prevent collisions, which leads to high communication delays. On the other hand, to disseminate data from the central node to the other nodes in the network, broadcast and/or multicast communications can be used. This imposes a very low bandwidth and latency overhead to the network. Moreover, all the sensor nodes in an automotive WSN are usually in the wireless communication range of the central node and can directly receive the actuation or setup commands from it, via broadcast messages. This is because high transmission powers can be used at the central node, as it has a wire connection with the main ECU and power consumption is not a concern for it. Accordingly, in this chapter, we focus on the data convergecast in automotive WSNs and the effect of network topology and TSCH scheduling on it.

The goal of this chapter is to minimize the average latency of data convergecast in an automotive WSN. In order to take advantage of parallel communications of TSCH to reduce data delivery latency, we propose the Low-Latency Topology management and TSCH scheduling (LLTT) technique. It is a cross-layer design which picks a proper network topology at the network layer to maximize the TSCH schedule utilization for the MAC layer. To match the extracted topology with the physical connectivity graph of the network, an optimized sub-graph isomorphism [28] algorithm is proposed. In the matching process, the quality of links and power plan of nodes are taken into account. The resulting network topology graph is used as the input of a light-weight TSCH schedule generator that minimizes the multi-hop communication latency. We moreover use particular shared timeslots for retransmissions, called grouped shared timeslots, in the schedule to increase the reliability of the multi-hop communications. Furthermore, a periodic aggregation technique is exploited to efficiently use the available bandwidth of multi-hop links.

LLTT has polynomial complexity and can be run on automotive ECUs with very low overhead and computation time. It also makes the TSCH schedule and accordingly behavior of the WSN predictable, targeting automotive applications with predefined Quality-of-Service (QoS) requirements. Experimental results show that LLTT achieves higher reliability compared to a star topology when latency bounds are defined for data validity. For high data generation rates, it provides lower average end-to-end data delivery latency, even compared to the TSCH minimal schedule [80].

This chapter is based on publication [77] and is organized as follows. An overview of the network management and TSCH scheduling techniques for dynamic and dense WSNs is given in the next section. The two main building blocks of LLTT, namely topology management and low-latency scheduling, are introduced in Sections 5.3 and 5.4. Section 5.5 introduces the exploited aggregation technique and gives an analysis of latency in the proposed mechanisms. Implementations and experimental setups are presented in Section 5.6. The achieved results are discussed in Section 5.7. Section 5.8 summarizes this chapter.

5.2 Network Management in Dense WSNs

Network topology and setup have a high impact on the performance of WSNs, especially for dense automotive networks. The authors of [61] show that under high traffic loads, congestion plays an important role in the performance of single channel WSNs. Authors propose to use multiple base stations to mitigate this problem. The authors of [39] show that using multi-hop networking for data convergecast in a vehicle improves the performance compared to single-hop convergecast. However, since a single channel and a contention-based MAC protocol is used in their work, multi-hop communications together with low transmission powers reduce the local contention in the network. This is the main reason for the observed performance gain. On the other hand, multi-hop communications increase the average transmission count, which leads to higher energy consumption. In [63], authors use extensive real-world experiments and show that an in-vehicle WSN can be separated into different zones, considering the behavior of wireless channels and link qualities. Thus, a clustering technique can be used to improve in-vehicle communication protocols.

In [58], Volvo group trucks technology presents a practical design of an in-vehicle WSN. They use the IEEE 802.15.4 TSCH protocol [7] as the MAC protocol for their experiments and show that this protocol is sufficiently robust to host low-to-medium time criticality. This robustness is due to the use of guaranteed communications and channel hopping. This work uses a network with only 10 nodes, while vehicles have the potential to use a much higher number of wireless sensors. For instance, a high end truck can have around 150 sensors. Assuming that 20% of this number can be migrated to short range wireless links, we would have a WSN with a node population of around 30 sensor nodes [57]. Compared to single channel protocols, TSCH is a better candidate for dense networks. This is because TSCH increases the network throughput by enabling parallel communications on multiple channels. This requires a network topology with links with different sources and destinations.

The TSCH scheduling task is an NP-hard problem [31]. Thus, all the available

scheduling algorithms use suboptimal solutions, targeting specific performance parameters. For instance, AMUS [44] is a centralized scheduling technique that reserves sequential timeslots for the set of links along an end-to-end route to reduce latency of multi-hop communications. This increases the scheduling complexity for dense networks in which the number of neighbors of each node is very high. [92] proposes a latency-optimal scheduling algorithm for convergecast in WirelessHART Networks. This technique gets the network topology as an input and, by taking advantage of parallel slotframes, it finds the shortest schedule for the communications. The authors show that the network topology affects the scheduling result and the end-to-end communication latency.

The distributed scheduling techniques such as Orchestra [25] and DeTAS [8] mainly focus on scalability and cannot guarantee a latency bounded schedule, because global network information is unavailable. However, some distributed scheduling techniques target low latency communications. For instance, Wave [69] aims to minimize latency by reducing the schedule length, using a repeated conflict-free schedule called wave.

All the scheduling algorithms that are proposed for the IEEE 802.15.4 TSCH protocol (e.g., [8, 13, 25, 30, 44, 56, 69]) require the network topology as an input and have no control on it. Thus, they cannot guarantee a predefined bound for the communication latency.

Network topology management is considered a task of the network layer. Topology management techniques typically aim to find a set of links to construct the network in order to provide energy efficiency [22, 42, 48, 82], delay bounds [47], and/or handling node failure [89]. To the best of our knowledge, there is no topology management technique that takes TSCH parallel communications into account, in order to increase the TSCH schedule utilization and reduce communication delay. As a cross-layer technique, LLTT manages network topology based on the number of nodes in the WSN and is able to provide a prior estimation of the resultant schedule and communication latency.

Using data aggregation for data collection in WSNs may follow a tree-based or cluster-based approach [29]. In both approaches, an aggregation point (e.g., cluster head) normally waits until it receives data from a set of active sources and then aggregates the data and forwards it to the sink node [79]. DICA [13] is a distributed data aggregation technique for single channel networks that uses data aggregation to employ the available bandwidth efficiently. This technique adjust the network tree formation to reduce latency for aggregation. In this chapter, we use a periodic aggregation technique in which a period equal to a TSCH slotframe is used to aggregate data in a subtree root. This technique manages the packet generation rate based on the available bandwidth in the TSCH MAC layer.

Allocating shared timeslots in a TSCH schedule for retransmissions to improve communication reliability is proposed in [40]. This technique also reduces the forwarding delay in a WSN. In the current work, we exploit shared retransmission timeslots and dedicate each one to a group of links in the network that have the same destination. This technique reduces the contention on accessing these timeslots by reducing the number of potential transmitters in each of them. Moreover, it enables allocating them in parallel over multiple channels.

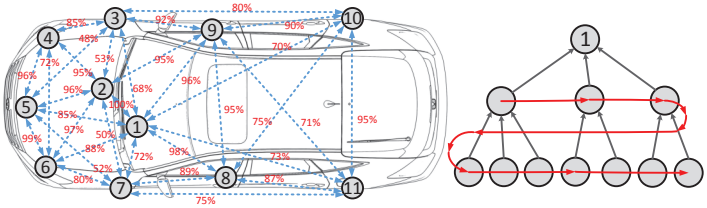
5.3 LLTT Topology Management

In this section, we introduce the topology management part of the LLTT technique. LLTT topology management consists of two steps. LLTT first selects a tree-topology structure T based on the number of available nodes in the network (right side of Figure 5.2(a)). This fixed tree structure is designed to provide high parallelism for the TSCH schedule, which leads to short slotframes. Moreover, it reduces the complexity of the TSCH scheduling algorithm, as the topology structure of the network is predictable. As a second step, LLTT topology management maps each node of V (the set of all nodes in the WSN) to a vertex in T (right side of Figures 5.2(a)-(e). In this process, link quality and the power condition of each node are taken into account. After selecting the best topology for the network, the scheduling part of LLTT (presented in the next section) schedules the defined links in the TSCH context.

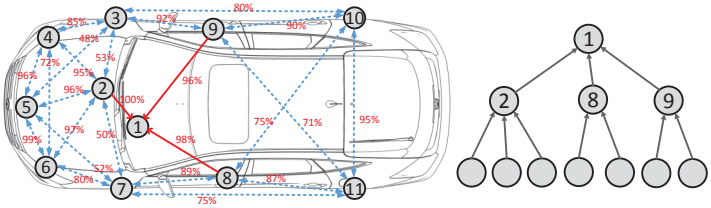
Two links $n_i \rightarrow n_j$ and $n_m \rightarrow n_k$ are independent iff $\{i, j\} \cap \{m, k\} = \emptyset$. This is because each node can only participate in one transmission or reception at a time. Two independent links can perform communications at the same time on different channels, while two dependent links should be serialized. Assuming a star automotive network, the sink node is one side of all communications. Thus all links are dependent, and parallel communication is not possible. To be able to maximally take advantage of TSCH parallel communications to reduce latency, the number of independent links in the network topology should be maximized. This requires the convergecast network to follow a multi-hop tree topology, rooted at the sink node. Accordingly, each link in a subtree is independent of all the links in other subtrees of the network (e.g., in Figure 5.2(e) all links towards node 2 are independent of links towards nodes 8 and 9).

While multi-hop communication provides more potential parallelism in TSCH scheduling, it increases the average transmission count in the network. This is because of packet forwarding that leads to higher energy consumption. Moreover, multi-hop communication may cause higher end-to-end packet error rates due to persistent interference. This requires retransmissions at each hop which comes with extra required timeslots per slotframe and thus higher data delivery latency. To lower these costs, LLTT limits the number of hops to two. LLTT constructs a two-level tree topology in which each nodes in the first level works as a router to forward the received data from its children to the sink node. Limiting the number of hops to two is in line with the fact that almost all wireless nodes in an automotive WSN are able to directly reach each other and a star network can satisfy the connection requirements.

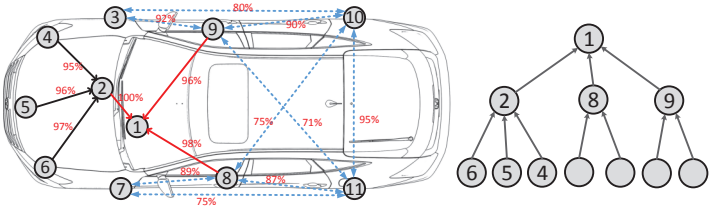
Assume that $ST^i \subset V, 1 \leq i \leq N_{ST}$ is the set of $|ST^i|$ sensor nodes of the i^{th} subtree, including the root of that subtree. N_{ST} is the total number of subtrees in the network. All the links in a subtree and the link from that subtree heading towards the sink node are dependent. Thus for ST^i, there are $|ST^i|$ links that should be scheduled in serial timeslots. This is equal to the degree of the root of ST^i. Also the N_{ST} links towards the sink node are dependent, which makes it necessary to allocate them into N_{ST} serial timeslots. Considering one slotframe to be allocated for links of each subtree, a higher number of nodes in each subtree leads to a longer slotframe required for it. This increases the time between data generation at end nodes and packet delivery at the sink node. A smaller N_{ST} leads to higher average $|ST^i|$, requiring longer slotframes and vise



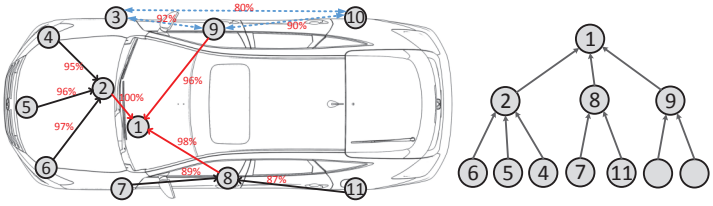
(a) Physical connectivity graph and topology graph



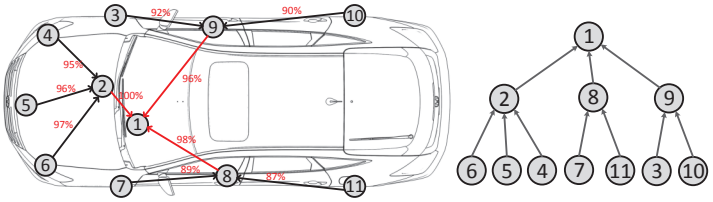
(b) Matching after 3 iterations, all subtree roots are matched first



(c) Matching after 6 iterations, children of node 2 are added



(d) Matching after 8 iterations, children of node 8 are added



(e) Matching after 10 iterations, children of node 9 are added

Figure 5.2: An example of matching in a small in-vehicle network with 11 nodes. Matching is done based on the quality of physical links in the network to build the network topology with high communication reliability. For simplicity of the example, all links are assumed to be symmetric.

versa. Thus, for all $1 \leq i \leq N_{ST}$, $|ST^i|$ should be as close as possible to the number of subtrees N_{ST}. This can be reached through a balanced complete k-ary tree. A balanced complete k-ary tree [70] is a rooted tree in which each node has k children, except the last level which can be incomplete as long as the distribution of leaf nodes is balanced.

We aim to find $k = N_{ST}$ based on the number of nodes in the network to build a balanced complete k-ary tree with height of two. As a balanced complete k-ary tree is a subgraph of a perfect k-ary tree with $k(k+1)+1$ nodes, the maximum number of nodes in a perfect k-ary tree with height of two is $k(k+1)+1$. The lower bound for the number of nodes in a balanced complete k-ary tree is equal to the number of vertices of the perfect (k-1)-ary tree.

$$k \times (k-1) + 1 < N \leq k \times (k+1) + 1 \tag{5.1}$$

Having N as the input, k can be calculated as

$$k = \lceil \frac{\sqrt{(4N-3)} - 1}{2} \rceil. \tag{5.2}$$

To assign independent links to parallel slotframes of a TSCH schedule, the number of available channels in the Hopping Sequence List ($|HSL|$) is the upper bound for k. If the number of nodes is higher than what a k-ary tree can support, we build the network topology with $|HSL|$ subtrees and divide the leaf nodes to all the subtrees equally. In the remainder, let T be the tree-topology structure derived in this way for a given problem instance.

After picking a proper tree-topology structure for the WSN, LLTT maps each node in the WSN to a vertex in that structure. This mapping is based on the physical connectivity graph of the network that captures link qualities. *Sub-graph isomorphism* [28] is used to perform the mapping. We enhanced the VF2 subgraph isomorphism algorithm [19] to reduce the memory usage. Considering the high connectivity between nodes in a dense WSN, this algorithms finds a match in polynomial time.

Algorithm 2 shows the high level functionality of LLTT topology management. Besides topology structure T, this algorithm gets the extracted link quality of each link in the network (LQ) and the power condition of each node (P) as its inputs. Each element of P holds a value between 0 and 1 based on the remaining battery percentage of the node (1 shows an wire-powered node). The algorithm maintains a connectivity matrix G. Each element of G gets a value of {*no_link*, *available_link*, *used_link*, and *blocked_link*}. It is initially filled with no_link and available_link values by applying a threshold to the LQ values. The algorithm further maintains a vector M of network nodes matched to T. It then starts from the root of T, which is assigned to the sink node of the network (node n_1 in Figure 5.2(a)). Then it goes through each vertex v of T, connected to already matched vertices, and finds a match in G.

Figure 5.2 shows some iterations of the matching technique for a simple network with $N = 11$ sensor nodes deployed in a vehicle. The physical connectivity graph (after applying a threshold of 50% on the link qualities) is depicted on the left side of Figure 5.2(a). For simplicity, the link quality of the two links between each couple of nodes is considered to be the same in this example. This figure also shows the balanced 3-ary tree (using Equation (5.2) and $N = 11$) extracted for the depicted network, rooted

ALGORITHM 2: Network subgraph isomorphism

Input:
T: tree-topology structure
LQ: $N \times N$ matrix that contains quality of the links between each pair of nodes in the network
P: vector of length N, contains power condition of each node
Output:
NT: network topology graph

```
1  M = ∅;
2  Initialize G from LQ;
3  FindMatch(G, LQ, T, P, M, n_1);
4  NT = ToTopology(T, M);
   /* Merge the tree-topology structure and the assigned node IDs to each vertex of it
      into a topology graph                                                          */
5  FindMatch(G, LQ, T, M, v)
6      if |M| = N then
7          return ;
8      else
9          v := NextVertex(T, v);
10         H[] := ∅ ;
           /* The history of nodes that are matched to v in this iteration        */
11         foreach u := NextMatch(G, LQ, P, M, H, v) do
12             add u to H[];
13             if IsFeasible(G, v, u) then
14                 AddPairedSet(G, M, v, u);
15                 FindMatch(G, LQ, T, P, M, v);
16                 RemovePairedSet(v, u, G, M);
17             end
18         end
19     end
20 end
21 NextMatch(G, LQ, P, M, H, v)
22     u' := M(parent(v ∈ T));
       /* The matched vertex of G to the parent of v in T                         */
23     find u ∈ G such that
24         1) (u ∉ M) ∧ (u ∉ H) ;
           /* u is not already matched                                            */
25         2) G(u, u') = available_link;
           /* There is a physical link from u toward its potential parent         */
26         3) degree(u) ≥ degree(v);
           /* The candidate u is at least connected with the same number of nodes that is
              required for connections of v in T                                  */
27         4) if(u' = sink) then P(u) = 1 is preferred;
           /* wire-powered nodes are preferred to be used as first level nodes    */
28         5) W(LQ(u, u'), degree(u), P(u), ...) has the maximum value;
29     end
30     return u;
31 end
```

at node n_1. The sequence of vertices for matching, which is defined by subroutine `NextVertex()`, is shown in Figure 5.2(a) as red arrows in the tree topology structure. This subroutine selects the subtree roots for matching first and then goes throughout leaf nodes in each subtree. This is because the link between each subtree root and the sink node is also used for forwarding data from leaf nodes towards the sink node and has a high impact on the reliability of the multi-hop network.

Using a history vector H, the algorithm keeps track of the candidates that failed to be matched in the current iteration. As shown in Algorithm 2, a set of rules is defined to select a candidate match for a given vertex of $v \in T$. The first three rules guarantee that u is a possible candidate to be matched to v. The fourth rule guarantees the use of wire-powered nodes in the first level of the topology graph, because they forward data from nodes in the second level and consume more power than the second level nodes. Any IEEE 802.15.4-enabled wireless node with wired connection towards the main ECU of the vehicle (e.g., gateway nodes) gets the highest priority to be selected as the first level node. The last rule selects the best possible candidate by assigning a weight to each of them. The weighting function **W** determines the suitability of a node to be matched to a vertex and can be calculated as

$$\begin{cases} (\alpha \times LQ(u,u') + \beta \times degree(u)) \times P(u)^2 & if \quad degree(v) > 1 \\ \alpha \times LQ(u,u') / (\beta \times degree(u) \times P(u)^2) & if \quad degree(v) = 1 \end{cases} \tag{5.3}$$

where coefficients α and β are to be determined by the user based on the range of link quality values and platform requirements. For instance, if the application requires very reliable links, $LQ(u,u')$ should be given a higher weight than the other parameter. If the number of wire-powered sensors in a network is less than N_{ST}, to satisfy rule 4, some of the first level nodes should be selected from the battery-powered sensors. The function **W** gives a high weight to the nodes with higher power levels to be matched to the first level nodes. By running this algorithm every once in a while (e.g., when the vehicle is going to be turned on), battery-powered sensors that are used as the first level nodes will change if there is another sensor available with higher battery charge. Re-running the algorithm also handles the long-term changes in the link qualities that are due to changes in the positioning of nodes, obstacles in the environment and the transmission power of nodes. Short-term changes in link qualities, that are mainly due to external interference, are handled by the TSCH interference mitigation technique that is proposed in Chapter 4.

At each iteration of the matching algoeithm, for the current vertex v and the selected matching candidate u, subroutine `IsFeasible()` checks whether by removing this node from G, and accordingly removing all its incoming links and its parent (if the parent has all its children matched), the graph G still remains connected. Otherwise, u is not an actual match for v in this iteration. If u is qualified to be matched to v, `AddPairedSet()` adds pair (v,u) to M and applies the required changes to G. These changes include:

1. changing the link between u and its matched parent from *available_link* to *used_link*;
2. if v is a leaf in T ($degree(v) = 1$), change all the *available_link* links to and from u in G to *blocked_link*;

3. if the degree of $Parent(v)$ in T is equal to the number of *used_link* links towards $Parent(u)$ in G, change all the *available_link* links to and from $Parent(u)$ in G to *blocked_link*.

After these changes, the connectivity graph G only keeps the links that still can be used for matching. Figure 5.2(b) shows the matching process after 3 iterations, when all parent nodes are matched. Figures 5.2(c)-(e) show the matching process after 6, 8, and 10 iterations where all children of each subtree root are matched. As depicted, after matching children of each subtree root, all the *blocked_link*s are removed from the possible candidate links for scheduling.

The algorithm proceeds to match the remaining vertices of T to the remaining vertices of G by recursively calling `FindMatch()`. By returning from each iteration of `FindMatch()` due to a matching failure, all the applied changes to the graph G in that iteration are restored by calling `RemovePairedSet()`. The algorithm finishes when a complete match ($|M| = |T|$) is found in the N^{th} recursion. Because this algorithm uses graph G (with four states for each link) to keep track of changes in each iteration, it requires a low amount of memory to execute. The match of T in G is used as the network graph NT, as input of the low-latency scheduling part of LLTT.

5.4 LLTT Scheduling

The output of the topology management part of LLTT is a two-hop tree network. In this section, we present the low-latency timeslot allocation for such a tree topology in detail. Algorithm 3 shows the process of low-latency scheduling. This algorithm allocates all the links in each subtree and the link from that subtree to the sink node to one slotframe with a unique channel offset for all timeslots. This is because the links in each subtree are dependent, while they are independent of the links in other subtrees. The links from subtree roots towards the sink node are also dependent and should not be parallelized. Furthermore, a link from a subtree root towards the sink node is dependent on the links in that subtree. Accordingly, multiple parallel slotframes can be used for links of different subtrees.

The required L_{SF} is given to the algorithm as an input. It can be calculated as:

$$L_{SF} = Max[degree(v) | v \in NT] + (2 \times N_{ReTx}) \tag{5.4}$$

where N_{ReTx} is the number of timeslots required for grouped retransmission in each subtree. The retransmission timeslots in the first and second levels of the network are dependent and cannot be parallelized with each other. Thus, the slotframe should be longer for twice the number of required timeslots for retransmissions in each level of the network. N_{ReTx} may be defined by the user or be extracted based on the communication statistics of the network. Figure 5.3 shows the schedule that is generated by this algorithm for an example network and $N_{ReTx} = 1$.

We define a *grouped retransmission timeslot* as a shared TSCH timeslot that is dedicated only for packet retransmissions of all links with the same destination. Assigning grouped retransmission timeslots in LLTT is based on the subtrees and is done by assigning a number of shared timeslots for possible retransmissions of all links in a subtree.

ALGORITHM 3: Low-latency schedule generator

Input:
L_{SF}: slotframe size given by Equation (5.4)
NT: network topology graph
N_{ReTx}: number of desired retransmission timeslots for each subtree
Output:
Schedule: $N_{ST} \times L_{SF}$ matrix, holds TSCH schedule

```
/* Allocate shared timeslots towards the sink node at the end of first slotframe    */
for i ← (L_SF − N_ReTx + 1) to L_SF do
    Schedule[1][i] := [SH, sink_node];
end
for sf ← 1 to N_ST do
    /* The timeslot for the link from the current subtree root to the sink node     */
    slot := L_SF − N_ReTx − sf + 1;
    /* Allocate a dedicated timeslot from the subtree root towards the sink node    */
    Schedule[sf][slot] := [root_of(ST^sf), sink_node];
    /* Allocate shared timeslots towards the subtree root right before the current
       timeslot                                                                     */
    for i ← 1 to N_ReTx do
        slot := slot − 1 ;
        if slot = 0 then
            slot := L_SF − N_ReTx ;
        end
        Schedule[sf][slot] := [SH, root_of(ST^sf)];
    end
    /* Allocate dedicated timeslots for the subtree right before the shared timeslots
       */
    foreach n_i ∈ ST^sf do
        slot := slot − 1 ;
        if slot = 0 then
            slot := L_SF − N_ReTx;
        end
        Schedule[sf][slot] := [n_i, root_of(ST^sf)];
    end
end
```

Each grouped retransmission timeslot can only be used to transmit packets that are not acknowledged in their dedicated timeslots.

The scheduling algorithm starts with allocating the last timeslots of the first slotframe for the grouped retransmission timeslots towards the sink node (the blue timeslot in Figure 5.3). Because all subtree roots are possible transmitters in these timeslots, these timeslots cannot be used for communications of all other links in other slotframes. Then the algorithm creates a slotframe for each subtree, using the already started slotframe for the first subtree. For each subtree, it allocates a timeslot for the link from that subtree root towards the sink node, together with the links in that subtree. Because these links are all dependent they are allocated to the same slotframe. Because of the

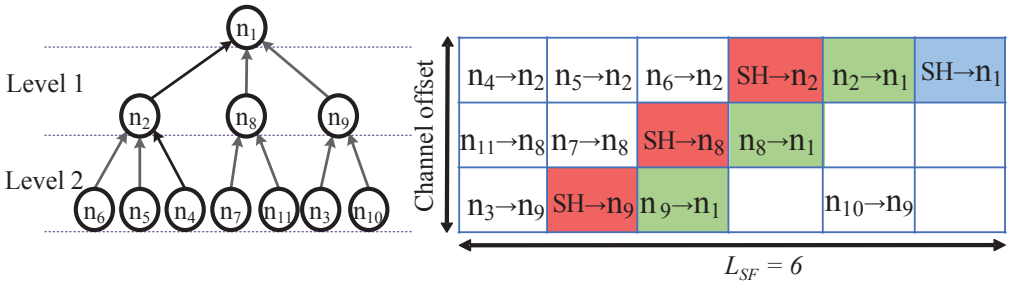


Figure 5.3: The TSCH schedule of an example network that is generated by Algorithm 3 (SH indicates shared slots).

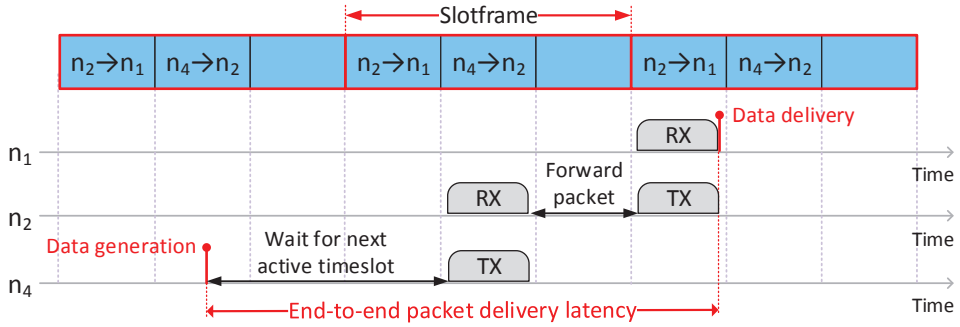


Figure 5.4: TSCH timeline of one packet transmission in a two-hop route from n_4 to n_1.

dependency between the links in the first level of the network, our algorithm uses different timeslots for the links from each subtree root towards the sink node (green timeslots in Figure 5.3).

Considering the example network in Figure 5.3, there is a two-hop path from n_4 to n_1 (sink node). This path requires two sequential TSCH timeslots to establish the end-to-end connection. As shown in the timeline diagram of Figure 5.4, the end-to-end latency is equal to sum of the buffering time at the source node (n_4) and the intermediate node (n_2). The data buffering time at n_4 may vary from 0 to L_{SF}, while The buffering time at n_2 depends on the placement of the allocated timeslots for the link in the second level and the following link in the first level within the TSCH slotframes. To reduce the buffering time at the intermediate nodes (subtree roots), the timeslots allocated for the links in a subtree should be before the timeslot allocated for the link from the root of that subtree towards the sink node. If the timeslot allocation algorithm reaches the beginning of a slotframe, it continues allocating timeslots for the subtree links from the end of the slotframe (link n_{10} to n_9 in the TSCH schedule of Figure 5.3), as slotframes repeat over time. Timeslots with already allocated dependent links are skipped in this process. This puts the allocated timeslots for the links in a subtree as close as possible to the timeslot allocated for the link from the root of that subtree towards the sink node in the next repetition of that slotframe.

To reduce the latency also for the failed transmissions during dedicated timeslots, grouped retransmission timeslots for the links in each subtree (red timeslots in Fig-

ure 5.3) are allocated between the dedicated timeslots for the second and first levels. Then the algorithm continues with allocating dedicated timeslots for all the links in that subtree, before the shared timeslots. Since this scheduling algorithm is based on a predefined topology structure, it has a low complexity compared to other central scheduling algorithms for TSCH networks.

The LLTT schedule provides guaranteed access to the medium for all links, targeting reliable and low-latency communications. This is done through parallel TSCH communications and a reduced slotframe size that increase the frequency of allocated timeslots to each link, supporting high data rates. A node in the second level of the network that has a lower data generation rate than the allocated bandwidth, can wake up and transmit in its allocated timeslots only if there is a packet ready for transmission in its MAC buffer. Thus, there is no power overhead for these nodes and low latency is still ensured. However, not all in-vehicle sensor nodes require low latency or high data rate communications. For those sensor nodes, a super-schedule can be used with a size of multiple LLTT slotframe sizes. The multiplication factor can be selected based on the data generation rate of the sensor node and the maximum allowed latency. This technique minimizes power consumption of the second level nodes and reduces the power consumption of the forwarding nodes at the first level, as they only listen to the allocated timeslot every multiple slotframes.

5.5 Data Aggregation and Latency Analysis

LLTT uses topology management and scheduling to provide a collision-free and low-latency TSCH schedule. Disabling retransmissions, the worst-case packet forwarding delay in an intermediate node can be expressed as the maximum time difference between the links in the second level of the two-hop tree and the following links in the first level (e.g., the time between Rx and Tx at n_2 in the timeline diagram of Figure 5.4). This delay, in terms of timeslots, is equal to L_{SF}. Since the timeslot duration is a constant in our work, we consider latency in terms of timeslots and do not convert it to time.

As sensory data is often very short, aggregation can be used at the intermediate nodes in the multi-hop paths to efficiently use the available bandwidth. When aggregation is used, all the data packets received from the second level nodes are aggregated with the subtree root's own data to be sent to the sink node. Aggregation reduces the number of timeslots in each slotframe needed by the nodes for forwarding data items received from others. This leads to more efficient use of available bandwidth of each link. Furthermore, it reduces the average communication latency due to reducing the size of the slotframe. We use a periodic aggregation at the network layer in periods of one slotframe. Shorter periods may cause overloading the MAC which increases the latency due to packet buffering after aggregation. On the other hand, longer periods lead to higher latencies due to longer buffering time for packets before aggregation.

The aggregation ideally should be done exactly before the allocated timeslot to the link from the subtree root towards the sink node, in order to lower the forwarding latency at the subtree roots. This requires a notion of TSCH MAC timing at upper layers (i.e., network layer), which is not practical. This periodic aggregation may lead to a delay of maximum L_{SF}, when a data packet is received exactly after the aggregation at

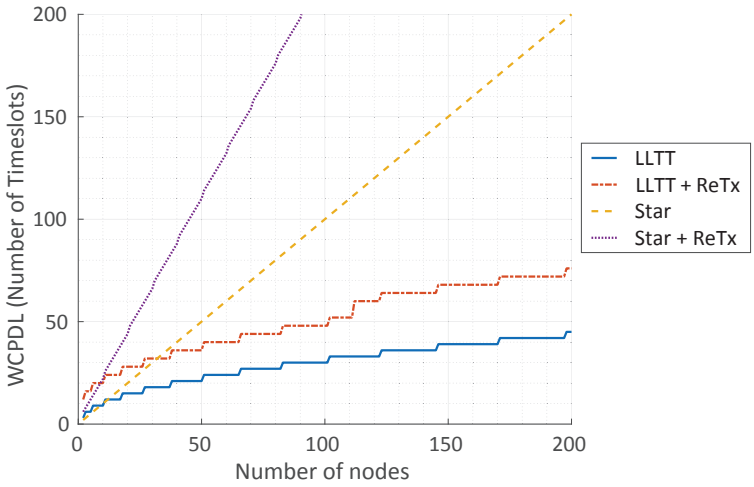


Figure 5.5: Analytical worst case packet delivery latency vs the number of nodes in the network for TSCH schedules of star and LLTT topologies.

the network layer. In this case, the data should wait for one slotframe to be put in the next packet. Thus, in total, the maximum packet forwarding latency in an intermediate node is $2 \times L_{SF}$.

Considering the data buffering time at the source node (see Section 5.4) and forwarding delay at intermediate nodes, the maximum packet delivery latency is $3 \times L_{SF}$. Using Equations (5.2) and (5.4), the Worst Case Packet Delivery Latency (WCPDL) (when retransmissions are disabled) is:

$$\begin{aligned} \text{WCPDL}_{No\_ReTx} &= 3 \times L_{SF} \\ &= 3 \times Max[degree(n) | n \in NT] \\ &\leq 3 \times (\lceil \frac{\sqrt{(4N-3)}-1}{2} \rceil + 1). \end{aligned} \tag{5.5}$$

Using retransmission timeslots to increase the reliability of the network leads to higher data delivery latency in a network. This higher delay has two reasons: 1) longer slotframes due to retransmission timeslots and, 2) the timeslot gap between the dedicated and retransmission timeslots for the links in the first level of the network. The retransmission timeslots in the second level are allocated in such a way so that they cause an extra latency of only one timeslot, due to the longer slotframes. This delay is therefore covered in the first point.

The dedicated timeslot for the link from the root of each subtree towards the sink node is the last point of data delivery when there is no retransmission. This is while with retransmission, the last chance for data delivery at the sink node is the allocated retransmission timeslot. This costs a latency equal to the gap between the dedicated and retransmission timeslots for the links towards the sink node which is at most equal to $L_{SF} - 1$. Accordingly, the WCPDL, when at maximum one retransmission is used, can

Figure 5.6: Experimental testbed with 30 JN5168 [53] wireless sensor nodes distributed over a car.

be extracted by Equation (5.6).

$$\text{WCPDL}_{ReTx} = 4 \times L_{SF} - 1 \tag{5.6}$$

Figure 5.5 shows the analytical WCPDL for star and LLTT networks for different numbers of nodes. As shown, WCPDL in a star TSCH network increases linearly with the number of nodes in the network, because all the links in the network are dependent. Using a two-hop network in LLTT, the WCPDL is proportional to the square root of the number of nodes in the network. This plot also shows the WCPDL when retransmissions are enabled, and 10% of the slotframe bandwidth is allocated for a maximum retransmission count of one. For a star network, due to the gap between a dedicated and shared timeslot for transmissions of a link (which is at maximum equal to the $L_{SF}-1$), retransmissions lead to a double latency compared to star with no retransmissions. LLTT with retransmissions can still provide a better latency bound compared to star with no retransmission when the number of nodes in the network exceeds 32.

5.6 Experimental Setups

We set up an experimental testbed of 31 NXP JN5168 dongles [53] in a car (Figure 5.6) for performance evaluation of LLTT. The transmission power of nodes is set to 0 dBm. The topology management and scheduling parts of LLTT are done centrally using Matlab on a host computer that is connected to the sink node (n_1) via a UART interface. The TSCH schedule is distributed by the sink node in an extra timeslot at the beginning of the first slotframe that is allocated for downstream data (using broadcasts). The sink node initiates the network as the PAN coordinator, and all nodes periodically generate data packets to be delivered to the sink.

To provide the LQ matrix for LLTT, initially, the network goes through an identification phase. In this phase, each node gets a dedicated timeslot and periodically broadcasts dummy packets. Accordingly, each node can extract quality of all its possible incoming links. We use PRR as the quality metric (LQI of incoming packets can be used

Table 5.1: Experimental setups and their slotframe size

Network	L_{SF}	**Comment**
LLTT	7	Schedule size of 6, using Equation (5.4) + One timeslot for EB advertisement
LLTT+ReTx	7+2	2 timeslots for retransmissions
Star	30+1	One dedicated timeslot for each node + One timeslot for EB advertisement
Star+ReTx	31+6	6 timeslots for retransmissions
Minimal	1	All timeslots active for Tx/Rx
Orchestra	31	sender-based slotframe

as well). After a predefined number of packet transmissions by each node (200 packets in our experiments), the sink node collects the results from all nodes and reports them to the host computer. Because this step is done once and only at the initialization of the network, the overhead is negligible.

For the used network with 30 sensor nodes and one sink node, the network topology is a complete 5-ary tree ($N_{ST} = 5$). We consider one third of the nodes as wire-powered sensors that are placed in different parts of the car. The rest are considered as battery-powered. The schedule generated by LLTT is broadcast to all nodes by the sink node. Then the main phase of the experiment starts, in which each node generates and sends periodic data packets.

As LLTT is a cross-layer design performing topology management as well as scheduling, we use some combinations of known networking and scheduling techniques for our performance comparison. We use a pure star network with a schedule that dedicates one unique timeslot to each node. In addition, we built two setups with combinations of RPL [88] and two TSCH scheduling mechanisms, namely the minimal schedule [80] and Orchestra [25]. For the TSCH minimal schedule, we use $L_{SF} = 1$ timeslot (slotted ALOHA), which leads to 100% duty cycle for all nodes. For Orchestra, we used the sender-based schedule, which was reported to have the best performance for convergecast [25], with $L_{SF} = 31$ and Enhanced Beacon (EB) slotframe size 53.

We perform experiments with retransmissions enabled and disabled for all the setups. We consider 20% retransmission bandwidth for each link, and accordingly decide on the number of shared retransmission timeslots for LLTT. For the minimal schedule, we conduct multiple experiments with different maximum number of link layer retransmissions from 1 to 3, and report the best results in terms of latency and reliability. To reduce contention on shared timeslots in the star network, we assign each shared timeslot to a group of 5 nodes, based on the retransmission bandwidth. This increases the size of the star slotframe by 6 when the network consists of 30 sensor nodes and one sink node. As Orchestra uses a hash of the nodes' MAC address to assign timeslots to nodes, there is a chance of allocating the same timeslot for multiple links. Because all nodes are in the range of each other and have a high data generation rate, this leads to interfered communications within the network and thus packet failures. The experimental results showed that only few nodes were able to deliver their data to the sink

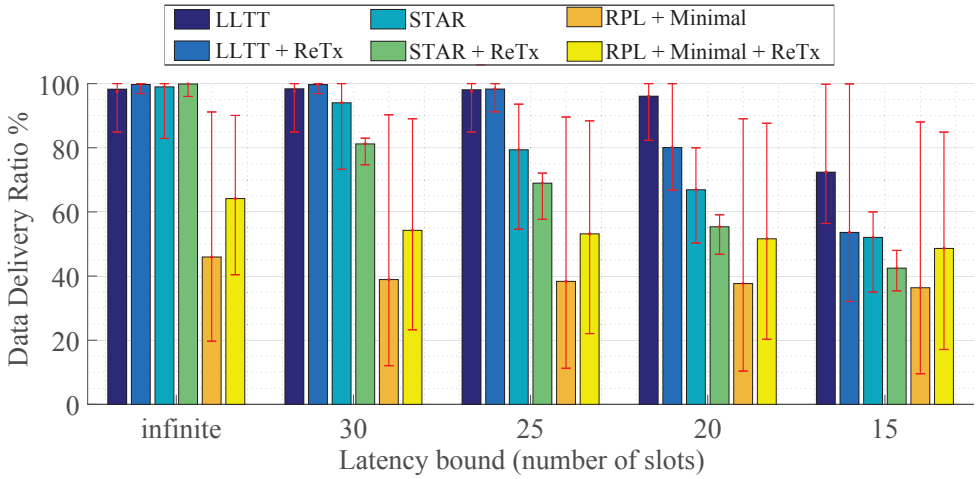


Figure 5.7: Average DDR of all links for different latency bounds in an interference-free environment. Red error bars show the minimum and maximum observed DDRs.

node. Thus, the results for RPL in combination with Orchestra were ultimately not considered in the performance analysis discussed in the next section. Orchestra is better suited for larger and lower density networks than the ones considered in this work. The network setups and their slotframe sizes are summarized in Table 5.1.

5.7 Performance Analysis

We investigate two metrics to evaluate the performance of LLTT, i.e., DDR and end-to-end latency. In the following, first we analyze the data delivery performance of LLTT, when different latency bounds are considered for validity of data by the application. Moreover, to evaluate the effect of shared retransmission timeslots, we performed a set of experiments in presence of controlled interference. Second, we investigate the performance of LLTT for different data generation rates.

5.7.1 Data Delivery vs Latency

Figure 5.7 shows the average DDR for different protocols and latency bounds, when the network is in an interference-free environment. A homogeneous data generation rate of 2Hz is used for all the nodes. Red error bars show the worst and best DDR for all 30 non-sink nodes in the network. When the applied latency bound is infinite so latency has no effect on data validity, both LLTT and star networks have a DDR higher than 98%. By using the retransmission timeslots, DDR is increased to 99.7% for both, while in the best case, RPL+Minimal has an average DDR of 64% with maximum retransmissions of one. For higher retransmissions, RPL+Minimal faces high traffic load of retransmitted packets, leading to even worse DDR and latency.

We use the results of the unbounded latency experiment and apply four latency

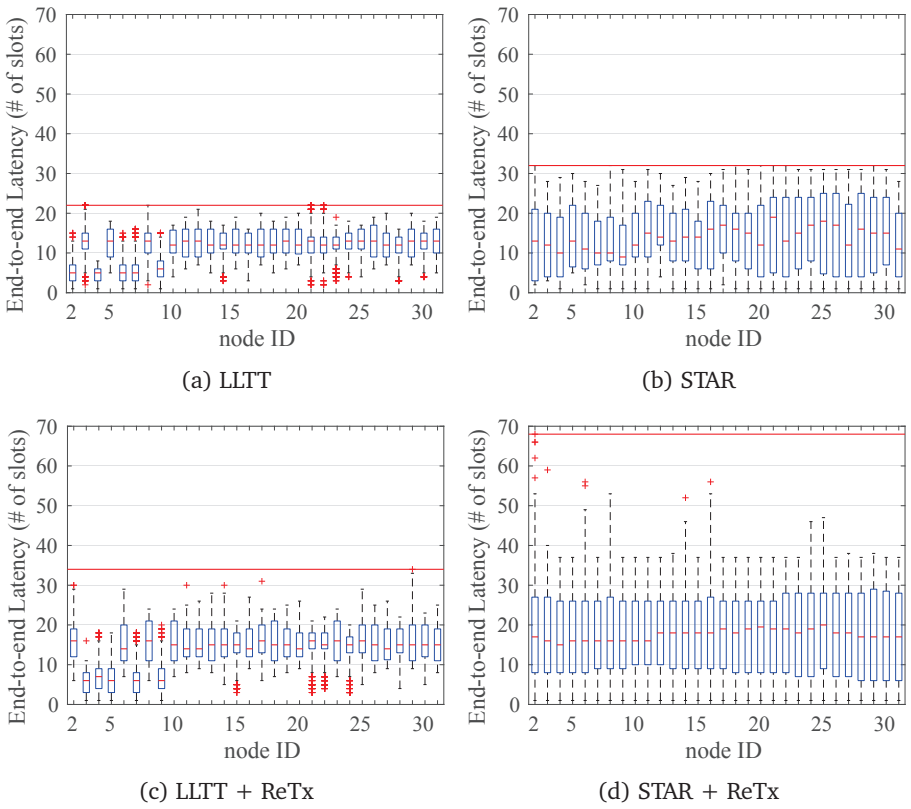


Figure 5.8: End-to-end latency distribution of each node's communication towards the sink node. X-axes show the node number and the red lines show the maximum observed packet delivery latency for all nodes.

bounds of 30, 25, 20, and 15 timeslots duration to it, i.e., 300, 250, 200, and 150 ms for a TSCH timeslot duration of 10 ms. Each bar in Figure 5.7 shows the percentage of valid data that is delivered to the sink node within the specified latency bound. As shown, by reducing the latency bound, there are less valid data packets delivered, especially for the star network. This is mostly because data has to wait in the MAC buffer of the source node for the next active timeslot to be transferred. This waiting time varies from 0 to L_{SF}. Because the star network has the longest slotframe, it causes higher average latency compared to others, and thus it is affected more by the latency bounds. This effect is more visible when retransmission slots are used, caused by the longer slotframes. LLTT reduces the buffering time at the source nodes by using very short slotframes. Thus, even for a latency bound of 25 timeslots, both versions of LLTT still deliver the same percentages of data packets as in the unbounded scenario.

Figure 5.7 shows that DDR of both versions of RPL+Minimal is almost the same for all latency bounds from 30 down to 15 timeslots. This is because all the timeslots

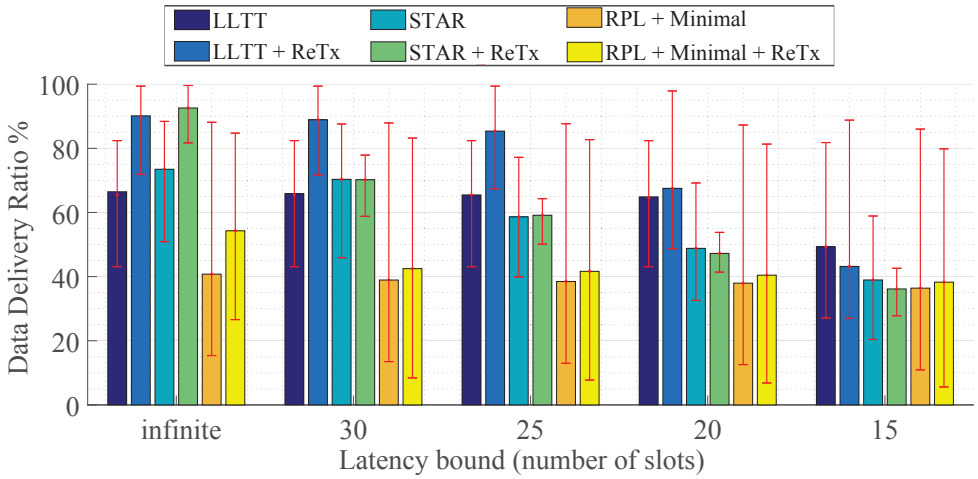


Figure 5.9: Average DDR for different latency bounds, in presence of controlled interference on 4 out of 16 channels.

are active for transmissions of all nodes and a generated packet at application layer can be transmitted at the same slot in the MAC layer. The DDR drop observed when going from unbounded to the 30 slot latency bound is due to the use of the back-off mechanism after transmission failures, which cause longer latencies that improve DDR in the unbounded scenario but invalidate data under latency bounds.

Figure 5.8 shows the achieved latency of each node for LLTT and star networks. The results confirm the analytical WCPDL (Section 5.5), considering one extra timeslot at each slotframe for network advertisement. This means that for a higher number of nodes than what is used in these experiments, LLTT even performs better than the star network. Another observation of Figure 5.8 is that five nodes in both LLTT experiments have lower latencies than other nodes. These are the subtree roots which have one-hop communications towards the sink node, while other nodes have two-hop communications. Accordingly, the in-vehicle network designer may decide to select the nodes with more stringent latency requirements as the subtree roots.

Retransmission timeslots are effective when physical layer links are not 100% reliable, otherwise they only lead to longer slotframes and higher latencies (Figure 5.8(c) and 5.8(d)). To evaluate the effect of the shared retransmission timeslots on communication reliability and latency, we conduct a set of experiments in presence of wireless interference. We used four interference generators to block 4 out of 16 frequency channels used for TSCH channel hopping by continuously generating dummy packets. Figure 5.9 shows the DDR results of these experiments for different latency bounds. Since 75% of the channels are not interfered, 75% of the links are expected to have successful communications. This is visible for the star experiment (with infinite latency bound), since all the communications are single hop. Due to multi-hop communications in LLTT and RPL+Minimal, the average DDR is lower. The high variation in DDR results of each experiment (red error bars) is because of difference in reliability level of one-hop and two-hop communications.

For all the experiments with retransmissions, we set the maximum retransmission count to one. As observed in Figure 5.9, grouped retransmission timeslots improve the average DDR about 24% and 20% for LLTT and star network, respectively. For RPL+Minimal, this improvement is less than 15%. This is because retransmissions impose more contention in the Minimal schedule.

A star network with retransmissions experiences a high drop in DDR when latency bounds apply. This implies higher average latency. This happens because failed packets should wait for shared retransmission slots. Longer slotframes due to addition of retransmission slots is another reason. Retransmission timeslots in LLTT are allocated in such a way so that they impose the lowest possible latency increase for single and multi-hop communications. For RPL+Minimal, the latency overhead of retransmissions is much lower, because all timeslots for transmissions and retransmission can take place after one back-off period. For very low latency bounds, the performance of the LLTT and star networks, which use dedicated timeslots, gets closer to the performance of RPL+Minimal with shared timeslots.

5.7.2 Effect of Data Generation Rate

To evaluate the performance of LLTT under different data generation rates, we picked five data generation rates (1, 2, 3, 5, and 10 Hz), and performed experiments for each network type with each data rate. No latency bound is considered for data validity. We performed these experiments in presence of interference generators. Since the slotframe size of both versions of LLTT is less than 10 timeslots (for 30 non-sink nodes) and aggregation is used by subtree roots, it can handle a data generation rate of 10 Hz (one packet every 10 timeslots). On the other hand, star uses a slotframe size of 31 timeslots and if more than one packet is injected to the TSCH MAC every 31 timeslots, latency increases dramatically. In order to prevent this, we use periodic local data aggregation in the star networks with a period of one slotframe. Higher packet generation rates for RPL+Minimal lead to higher contention rates and lower performance. This decreases DDR when retransmissions are disabled, and increases latency when retransmissions are enabled. We use the periodic aggregation technique for RPL+Minimal with a period of 3Hz. This reduced contention helps more packets to be delivered per time unit, but with a higher average latency.

Figure 5.10 shows the average achieved DDR and latency in each network under different data rates. This figure reveals that both versions of LLTT have almost the same average end-to-end latency for different data rates. For RPL+Minimal, the achieved latency increases when increasing the data generation rate. At the same time, average DDR goes down. This is due to higher contention which increases failure rate as well as the waiting time for packet (re)transmissions. While aggregation only affects the packet size and has no effect on the contention rate, for data rates higher than 3Hz, the achieved DDR is almost the same for both RPL+Minimal setups (with and without retransmission). In contrast, periodic aggregation increases the average latency with about half the aggregation period, i.e., 16 timeslots, for data rates higher than 3Hz. The same happens for the star setup when the data generation period is lower than L_{SF}. For star without retransmission timeslots, this happens for data rates higher than 3Hz with a latency increase of around 16 timeslots ($L_{SF} = 31$). This is while for star

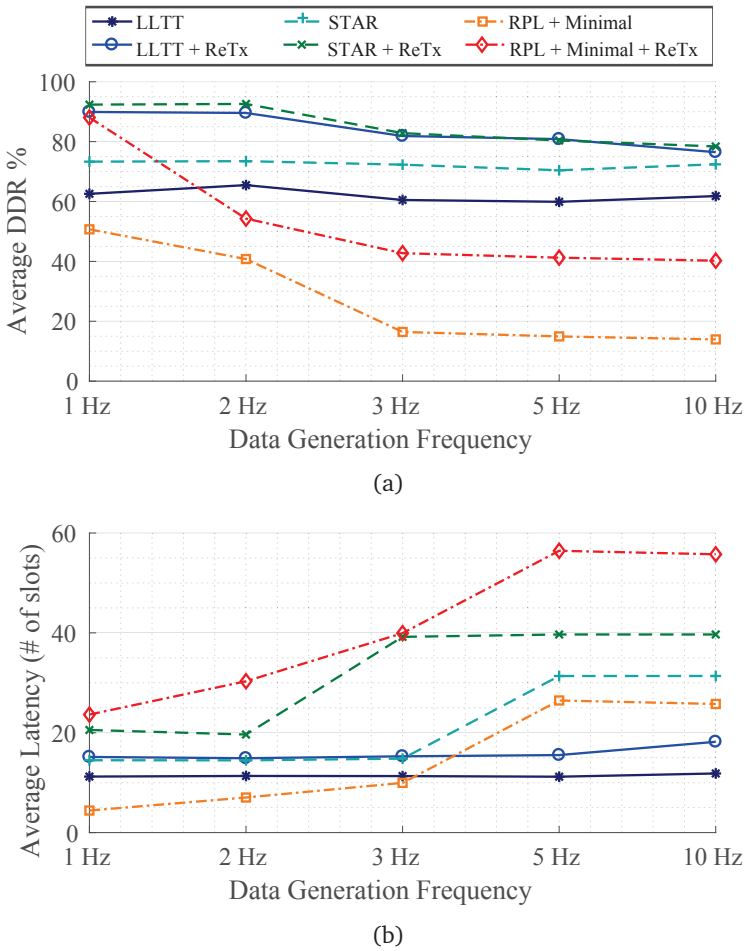


Figure 5.10: a) Average DDR and b) average data delivery latency for different networks and data generation rates, in presence of interference generators.

with retransmission timeslots, this effect is observed for data rates higher than 2Hz with about 19 timeslots latency increase ($L_{SF} = 37$).

As all the communications of LLTT and star use dedicated links, they show almost constant DDR for all data rates. When shared retransmission slots are exploited, lower data rates lead to higher DDR. This is because of the lower contention on shared retransmission timeslots that increases the chance of packet delivery for failed transmissions on dedicated timeslots. For low data generation rates, LLTT with retransmission timeslots provides similar DDR to the star and RPL+Minimal networks with retransmission, while it has over 25% lower average latency. Furthermore, for high data generation rates, LLTT outperforms star with almost 60% and RPL+Minimal with 70% lower average latency. It shows the ability of LLTT to provide reliable and low latency data communications for a dense network with high communication loads.

5.8 Summary

This chapter focused on communications in dense TSCH-based automotive wireless sensor networks in which all sensor nodes send their data to a single sink node in the network. The goal is to reduce the latency of end-to-end communications to enable more potential retransmissions within the defined latency limits, and accordingly increase the reliability of communications. Accordingly, this chapter presents Low-Latency Topology management and TSCH scheduling (LLTT), a cross-layer design for reliable and low-latency data convergecast in dense TSCH-based networks. Based on the number of nodes in the network, LLTT picks a balanced two-level complete k-ary tree as the topology structure of the network. This tree-topology structure provides a high number of independent links, which paves the way for a high degree of parallelism in TSCH schedules. Employing an optimized graph isomorphism algorithm, LLTT extracts a proper match for the selected tree structure in the connectivity graph of the network. The extracted network topology is used by a light-weight TSCH schedule generator to find a highly parallelized TSCH schedule. This schedule generator allocates grouped retransmission timeslots among dedicated timeslots to improve communication reliability. Periodic data aggregation is used to improve bandwidth utilization. Experimental results in an in-vehicle testbed with 31 sensor nodes and in presence of interference generators show that LLTT is able to provide latencies below 250 ms, while it keeps communications reliable by using retransmissions. Experiments show that even for high data generation rates, the periodic aggregation technique of LLTT keeps the average communication latency low.

The proposed mechanism in this chapter used a static TSCH schedule for the automotive WSNs. Dedicating timeslots to the links improves the reliability of the network by preventing internal collisions. This allocates a constant bandwidth to each link, as timeslots repeat over time with a periodic pattern. However, due to the spatial and temporal heterogeneity of automotive WSNs, the required bandwidth by each link may change over time. Allocating the worst-case bandwidth requirement to a link may lead to shortage of communication resources and affect the functionality of the network. On the other hand, a constant bandwidth allocation leads to waste of communication resources when a link requires less bandwidth than the allocated amount. Accordingly, there should be a mechanism to support this heterogeneity in automotive WSNs and efficiently manage the available communication resources over time. We address this problem in the next chapter by proposing a new timeslot type for the TSCH protocol, which enables on-demand sharing of dedicated timeslots.

6

Heterogeneity Support in TSCH Networks

6.1 Overview

Heterogeneity is a native property of automotive WSNs. It exists in different aspects such as the operation mode of applications, data generation patterns, reliability requirements, and medium conditions. For instance, automotive WSNs connect several types of sensors in a vehicle to a central entity, each type running a different application. This heterogeneity and the resulting dynamics should be properly supported by the network protocol stack. Otherwise, it is not possible to guarantee Quality-of-Service (QoS) for the network, and through this its dependability.

The IEEE 802.15.4 TSCH protocol defines two types of timeslots for communications, namely dedicated and shared timeslots. An upper layer in the protocol stack uses these timeslots to design a communication schedule for the network links, based on the required bandwidth for each link. Considering a network with time-varying data traffic generation by each node, the bandwidth requirements are changing over time for each link. This leads to waste of resources when applications are running in non-worst-case modes. The unused resources in this situation could be used for communications of other applications by defining proper mechanisms. On the other hand, using shared resources may lead to high contention, when multiple applications are transferring large data volumes.

In this chapter, we propose a hybrid timeslot that can be used as a dedicated timeslot for communications by one owner node and as a shared timeslot for all other nodes to transmit data to the same destination, if the owner node does not use it for transmission. This is done by performing one or more Clear Channel Assessments (CCAs) by the non-owner nodes with a *small delay*; this allows those nodes to detect whether the owner node skips transmission in that timeslot. The same TSCH Carrier-Sense Multiple Access with Collision Avoidance (CSMA-CA) retransmission algorithm that is used for the shared timeslots in the IEEE 802.15.4 [7] protocol is used to manage contention in accessing the hybrid timeslots. The proposed hybrid timeslot imposes very little change

to the TSCH protocol and has backward compatibility with it. However, the overhead of this technique, compared to the basic TSCH protocol, is the need for a little longer timeslots or shorter maximum size of those packets that are transmitted in the hybrid timeslots by the non-owner nodes. The reason is that transmissions of non-owner nodes in a hybrid timeslot start with a small delay due to the delayed CCA.

We performed a set of experiments and simulations to evaluate this technique. Results show that using hybrid timeslots in a TSCH schedule leads to lower communication latency compared to only using dedicated or shared timeslots. Moreover, it improves the reliability of the network by reducing the number of packet drops due to buffer overflow. Moreover, the average power consumption of packet delivery by using hybrid timeslots is lower than that of shared timeslots.

The chapter is based on publication [76] and is organized as follows. Related work about handling traffic heterogeneity in TSCH-based WSNs is surveyed in Section 6.2. The design details of the hybrid timeslots are presented in Section 6.3. Performance evaluation setups and results are discussed in Section 6.4. Section 6.5 summarizes this chapter.

6.2 A Review of Handling Traffic Heterogeneity

The IEEE 802.15.4 TSCH protocol defines two generic types of timeslots, namely dedicated and shared timeslots. Dedicated timeslots are defined to be exclusively assigned to a specific [transmitter, receiver] couple, called a link. This guarantees that only one node is transmitting on a specific [timeslot, channel] and there is no interference from nodes in the network for that communication. Shared timeslots are defined to share the medium between multiple source-destination nodes through a slotted CSMA-CA mechanism. These timeslots are usually used for communications with low bandwidth and low reliability requirements. The IEEE 802.15.4 standard leaves the scheduling task to the upper layers in the protocol stack (i.e., a sublayer between network and MAC layers). Such a scheduler decides on the number of dedicated and shared timeslots for each link, based on the application data rate and its QoS requirements.

Time variations of application's specifications and requirements makes the scheduling task very challenging. This is because every change in the application data rate and its requirements may need changes in the TSCH schedule. On the other hand, changes in the channel quality can affect the effective allocated TSCH bandwidth to a link, and require the TSCH schedule to be adapted accordingly. [40] presents a scheduling algorithm that uses shared timeslots for retransmission of different flows in order to satisfy required reliability. This shortens the slotframe size and forwarding delay by reducing the number of required dedicated timeslots in each slotframe. Besides using shared timeslots next to the dedicated timeslots for a link, [27] introduces slot reuse for the forwarding nodes, by which dedicated timeslots to each node's indirect children may be used by the node for upstream data forwarding. Results shows better reliability than simple timeslot sharing in multihop topologies. However, these techniques may lead to waste of MAC bandwidth if the data generation rate of an application becomes less than the dedicated bandwidth allocated to it.

On-the-fly bandwidth reservation [55] is a dynamic scheduling technique that aims

at adapting the TSCH schedule of a node to its actual bandwidth requirements. This technique constantly monitors the amount of data being sent towards each of the node's neighbors. Then if the data rate changes, it asks the upper sub-layer (i.e., 6top [83]) to add or delete dedicated timeslots to the schedule. This technique requires continuous monitoring of application data traffic and negotiation between neighbor nodes, which results in communication overhead. It imposes continuous changes to the TSCH schedule and has a delay of few slotframe periods to apply required changes to the TSCH schedule. Furthermore, authors use a constant slotframe length for this setup that may not satisfy the latency requirements of some applications.

The available dynamic scheduling techniques (e.g., [40], [27], and [55]) dedicate an initial bandwidth to each link and use different techniques to adapt it over time based on the changes in the application and/or channel behavior. However, these scheduling techniques can only use dedicated and shared timeslots that are either available to only one link or to all links. If a dedicated timeslot in a schedule is not used for transmissions by the assigned link, no other link is allowed to use it until it is removed from the schedule by the scheduler. The slot reuse technique that is introduced in [27] is only operational for links under the same routing hierarchy. On the other hand, a shared timeslot can be potentially used by all links. If more than one link in a neighborhood uses it for transmissions, all communications fail with a high chance. These restrictions inspired us to design hybrid timeslots for the IEEE 802.15.4 [7] TSCH MAC to be used by different schedulers to control the dynamism of the bandwidth requirements at the timeslot level. Using this new type of timeslot, a scheduler can only consider an average amount of bandwidth to be allocated for each link and there is no need to add or remove timeslots at runtime to handle variations in the links' bandwidth requirements. In other words, hybrid timeslots take care of dynamic bandwidth requirements in a heterogeneous WSN.

6.3 Hybrid Timeslot Design for IEEE 802.15.4 TSCH

6.3.1 Background

Our proposed idea of using hybrid timeslots that act as both dedicated and shared timeslots is inspired by the Z-MAC [64] protocol. Z-MAC assigns two types of transmitters to a communication slot, namely owner and non-owner transmitters. Based on the type of the transmitter in each slot, Z-MAC performs a random back-off within a first (for owner) or second (for non-owners) contention window. After the back-off period, it runs CCA and if the channel is clear, then it starts transmission. This gives higher priority to the owner transmitter of a slot compared to the non-owner transmitters. Z-MAC uses CSMA-CA in each slot and thus, the slot size must be larger than the sum of the two contention windows, the CCA period and one packet propagation time. However, the IEEE 802.15.4 [7] TSCH protocol considers no contention period within a timeslot in order to reduce the timeslot size and increase the bandwidth utilization. Instead, TSCH uses back-off on timeslots to handle contentions on accessing shared timeslots.

While Z-MAC uses message exchanges to negotiate on the owner transmitter of a timeslot, we add hybrid timeslots to the TSCH protocol as a new type of timeslots that a

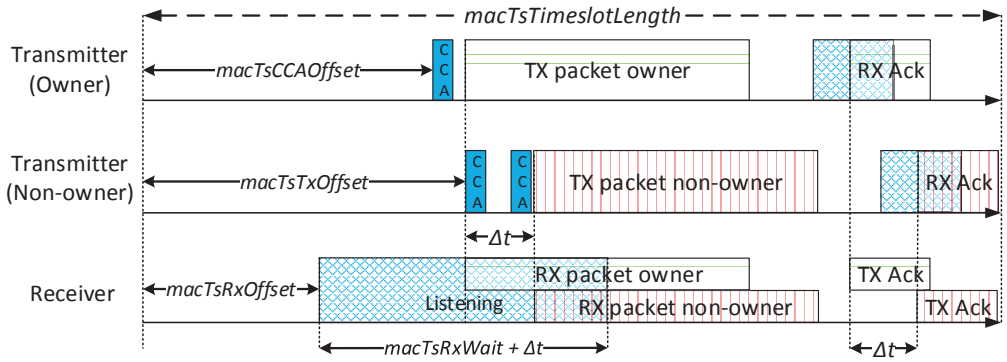


Figure 6.1: Timeslot diagram for a hybrid timeslot. An owner transmitter follows the default timeslot diagram, while a non-owner transmitter uses a Δt delay for communication offsets.

TSCH scheduler can use for scheduling. We design a diagram for hybrid TSCH timeslots that follows the basic TSCH timeslot diagram. This diagram enables defining owner and non-owner access to a timeslot. Only one owner node can be assigned to a hybrid timeslot to have guaranteed access to it. We dedicate a part of hybrid timeslots for performing CCAs by the non-owners to check if the owner is active. Thus, the non-owner nodes can use that timeslot for shared transmissions (as defined in the TSCH protocol), whenever the owner skips packet transmission in that timeslot.

6.3.2 Design

Figure 6.1 shows the proposed diagram for the hybrid timeslots. It has the same timeslot timeline as the one defined in the IEEE 802.15.4 TSCH [7] standard for the owner node, whereas it has a different timeline defined for communications of non-owner nodes in the hybrid timeslots. In this timeline, a non-owner transmitter wakes up at *macTsTxOffset* offset from the beginning of the timeslot. Then, instead of transmission, it listens to the medium for a Δt period. This listening period is to detect if the owner of the timeslot starts a transmission in this timeslot. We use CCAs with mode 3 to do this. This CCA mode reports a busy medium if it detects a signal with the IEEE 802.15.4 modulation and spreading characteristics or signal energy above a threshold. Using only one CCA in a very short Δt period may lead to missing the owner transmission due to synchronization error between timeslots of the owner and non-owner nodes. We propose to perform two CCAs, one at the beginning of the Δt period, and one at the end of it. The first CCA guarantees that if the timeslot of the non-owner starts later than the owner node and the owner transmits a very short packet, the non-owner still will be able to detect that packet transmission. Otherwise, the non-owner transmission might then clash with the acknowledgment of the owner packet. Placing the second CCA at the end of the Δt period helps a non-owner to detect transmissions of the owner even if the timeslot of the non-owner is ahead of the timeslot of the owner. If both CCAs show a clear channel, a non-owner considers this as skipped transmission by the owner and

starts packet transmission at *macTsTxOffset*+Δt offset. Accordingly, the acknowledgment process is delayed by Δt.

The Δt duration should be defined based on the timeslot synchronization error margins. As defined in the IEEE 802.15.4 TSCH [7] standard, a receiver node wakes up earlier than the *macTsTxOffset* for a *guardtime* (typical value of $1ms$, considering preamble transmission time [75]). This is done to compensate synchronization errors when the timeslot of the transmitter is ahead of that of the receiver. Also, the receiver continues listening for a *guardtime* after the *macTsTxOffset*, to compensate backward synchronization error. Therefore, a *guardtime* is defined to be the maximum synchronization loss between two nodes in the network. Accordingly, an owner and non-owner of a hybrid timeslot may lose synchronization for a *guardtime*. To enable a non-owner node to detect packet transmissions by the owner, even in the case that it starts its timeslot ahead of the owner, we need to set $\Delta t \geq$ *guardtime*. This enables the second CCA in the Δt period to still detect transmission by the owner, if the non-owner timeslot is ahead. We choose the Δt = *guardtime* to impose the least overhead to the TSCH timeslots.

If the owner skips transmission in a hybrid timeslot, the receiver should keep listening for a longer time to receive a packet from one of the non-owner nodes. As the *Tx Offset* is delayed for a Δt in the timeline of non-owner transmitters, the listening phase at the receiver node should be extended for a Δt period to compensate that delay. The default listening duration is defined as *macTsRxWait* in the protocol, which is equal to twice the synchronization loss *guardtime* plus the preamble transmission time. Thus, the receiver in a hybrid timeslot shall listen for the start of an incoming packet for a longer time equal to $3 \times$ *guardtime* plus the preamble transmission time.

Since multiple non-owner nodes may try to use a hybrid timeslot for their transmissions, collisions may happen. This is the same situation that happens in the shared timeslots. Thus, we treat transmissions by the non-owner nodes in a hybrid timeslot the same as the shared transmissions and use the same CSMA-CA algorithm specified by the IEEE 802.15.4 [7] TSCH protocol for shared timeslots.

All the dedicated timeslots in a TSCH schedule can be replaced by hybrid timeslots, considering the dedicated transmitter as the owner of the timeslot. This can share the unused bandwidth that is dedicated to each node with other nodes. This enables more potential retransmissions to be done within the latency limits to increase reliability of communications. Also, this reduces the average packet delivery latency by reducing buffering time at the source node. This also reduces the need for shared timeslots, that are normally used in a TSCH schedule for retransmission of un-acknowledged packets, resulting in shorter slotframes. Shorter slotframes lead to higher allocated bandwidth to each link and lower communication latencies. Accordingly, by replacing dedicated timeslots with hybrid timeslots in the LLTT technique (proposed in Chapter 5), the performance improves in both reliability and latency terms.

Adding hybrid timeslots to a TSCH MAC imposes no special adaptation to the IEEE 802.15.4 TSCH standard and is backward compatible. This means that the nodes with hybrid timeslots enabled and nodes without ability to use hybrid timeslots can communicate without problems within the same network. This only requires increase of the *macTsRxWait* duration by Δt.

6.3.3 Design Trade-offs

A non-owner transmitter in a hybrid timeslot may aim at sending the packet to either the same destination as the owner, or a different one. If a non-owner node aims to transmit its data packet to a different node, that receiver should be aware of this decision and listen in that timeslot. As multiple non-owners may share the same hybrid timeslot, multiple receivers should be listening in each hybrid timeslot to receive packets from multiple possible sources. For each transmission by the owner or non-owner nodes, all the receivers should receive that packet and then check if the packet is for them. This imposes a considerable idle listening and overhearing energy wastage. Furthermore, coordinating transmitters and receivers on using a hybrid timeslot in the non-owner mode adds overhead to the TSCH scheduler. Accordingly, we recommend that in a hybrid timeslot, a non-owner transmitter only uses the timeslot if it has data towards the same destination as the owner.

Using a common receiver for hybrid timeslots fits well with the idea of grouped retransmissions and tree topology structure that are used by LLTT (Chapter 5). This is because all nodes in each subtree of LLTT send data to a common receiver and they can share hybrid timeslots with each other. This also prevents cumulative clock drifts between owner and non-owners of a hybrid timeslot, as they are synchronized to the same parent. This guarantees the correct functionality of hybrid timeslots.

The delayed communications of non-owner nodes in hybrid timeslots require a Δt extra time within a timeslot. This can be reached either by increasing the length of all timeslots by Δt, or reducing the maximum size of the packets that get transmitted in a hybrid timeslot by the non-owner nodes. Using longer timeslots leads to an overhead for ordinary timeslots, while the second technique does not have such an overhead. By using the second technique, a non-owner node can only use a hybrid timeslot for transmission if the size of its packet is short enough to be transmitted within the timeslot bounds. A maximum size packet in the standard (133 bytes in the physical layer) takes $4256\mu s$. This time should be reduced to $(4256\mu s - \Delta t)$ for non-owner transmissions in hybrid timeslots. Accordingly, the maximum length of the packet can be calculated in bytes (one byte per $32\mu s$). In typical automotive applications in which a WSN is used for monitoring, application data is usually only a few bytes. This gives the opportunity to all data packets to be transmitted in hybrid timeslots. However, nodes are still able to transmit the maximum size packets in the ordinary timeslots, and hybrid timeslots used by the owner. Transmission of the maximum size packets is necessary to handle protocol-defined packets (e.g., EB packets). If the network is required to support the maximum packet size that is defined in the protocol for all packets, the timeslot size needs to be increased by Δt.

6.3.4 Hidden Terminal Problem

The hidden terminal problem may affect the functionality of hybrid timeslots. This happens when two-hop neighbors of an owner of a hybrid timeslot try to send packets on that timeslot as non-owners. In this case, they cannot detect the transmission by the owner and thus, they use the timeslot for transmission. This may cause packet reception failure at the receiving node. However, for small networks such as in-vehicle networks,

this situation never happens, as all wireless nodes are in the range of each other. For larger networks, there are multiple options to prevent this problem. One option is to consider this hidden terminal problem during TSCH scheduling. This can be done by assigning non-owner transmitters to a hybrid timeslot only if they are a direct neighbor of the owner. Another solution is that the scheduler only takes care of assigning owner transmitters to the hybrid timeslots in the same way as for dedicated timeslots. In this case, at runtimen each node can broadcast a message to all of its one-hop neighbors, specifying the hybrid timeslots that it owns. By receiving this message, a node can add those hybrid timeslots to its schedule, as non-owner timeslots. Thus, only the one-hop neighbors of the owner node that are able to detect whether or not the owner node skips transmission on a hybrid timeslot can use it as non-owner nodes. This prevents the hidden terminal problem.

6.4 Performance Evaluation

6.4.1 Setup

To evaluate the functionality and performance of hybrid timeslots within a TSCH schedule, we added it to the TSCH protocol implementation on top of the Contiki [24] operating system. We perform a set of lab experiments using 10 NXP JN5168 dongles [53]. We deploy a star network with one coordinator (node 1) and nine sensor nodes that send packets to the coordinator. Each node is the owner of one hybrid TX timeslot in a slotframe of size $L_{SF} = 10$, with the coordinator as destination. The first timeslot of the slotframe is used for the network advertisement by the coordinator. Other nodes can use hybrid timeslots as non-owner transmitters, if they have waiting packets to transfer.

To better investigate the performance of the hybrid timeslots in comparison with other types of timeslots, we also perform a set of simulations using COOJA [54]. We use the same network setup as the one used for the lab experiments, using Sky motes that emulate the behavior of the TelosB/Tmote Sky platform [49]. In our simulations, we study the effect of physical layer reliability on the performance of different schedule types. Due to the timing limitations of the Sky platform, timeslots with length $15ms$ are used instead of default length $10ms$. To have consistency between the results of experiments and simulations, we also use timeslots with length $15ms$ for our lab experiments.

Considering a *guardtime* of $1ms$ for the TSCH timeslots, hybrid timeslots are required to be $1ms$ longer or have about 32 bytes shorter maximum size of packets. Accordingly, we define two types of schedules for hybrid timeslot evaluations. One, the Hybrid schedule, contains hybrid timeslots with length $15ms$ and physical frame size up to 101 bytes. The other schedule type is called L-Hybrid and uses hybrid timeslots that are $\Delta t =$ *guardtime* longer in length ($16ms$) and can handle the default maximum size frames.

Hybrid timeslots can be used by any type of TSCH scheduler or bandwidth control mechanism. Thus, for performance evaluations, we compare the performance of this new type of timeslot with the dedicated and shared timeslots. Accordingly, we define two other TSCH schedules. The first schedule consists of a slotframe of size 10 timeslots in which each timeslot is dedicated to one node to send its packet to the coordinator. The other schedule has only one timeslot that is shared between all nodes for transmission and reception (Minimal schedule with $L_{SF} = 1$) .

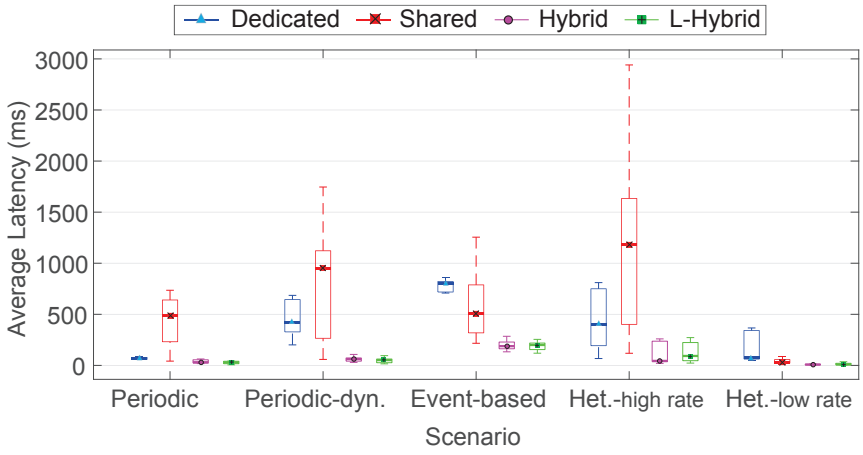


Figure 6.2: Distribution of links' average packet delivery latency for different schedule types and different data generation patterns.

In our experiments, we extract the performance of each schedule under different data generation scenarios, namely periodic, dynamic-periodic, event-based, and heterogeneous. In the periodic scenario, each node uses a fixed period for data generation. As node-id for non-coordinator sensor nodes starts from 2, we use multiplication of the node-id of each node and half of the slotframe length as its data generation period. For the dynamic-periodic scenario, every two seconds a random data generation period between 0.5 and 8 slotframe lengths is selected for each node. In the event-based data generation scenario, each node sends 10 packets in a burst (one packet every timeslot) after a random time between 2 to 4 seconds. This process repeats over time. We use a combination of the three data generation scenarios as the high-rate heterogeneous scenario (Het.-high rate), in which every three transmitter nodes use one of the data generation patterns. Furthermore, we define a low-rate heterogeneous scenario (Het.-low rate) by reducing the data generation rate of each node to 10% of the high-rate heterogeneous scenario.

To have a clean comparison between different timeslot types, we place all the sensor nodes in an interference-free environment and with a short distance of each other. This provides fully reliable links for all the experiments. The maximum retransmission count of the MAC layer is set to 6. The size of the MAC outgoing buffer towards each neighbor is set to 16 packets.

6.4.2 Experimental Results

In our lab experiments, we investigate the performance of different schedule types under different data generation scenarios. Figure 6.2 uses boxplots to show the average packet delivery latency over the nine available links, for all the schedules and scenarios. This figure shows that the longer timeslots of the L-Hybrid schedule lead to a little

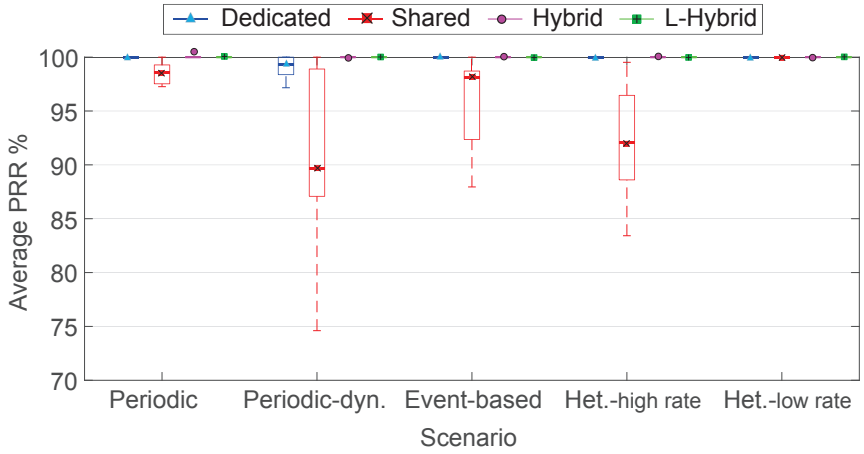


Figure 6.3: Distribution of links' average PRR for different schedule types and different data generation patterns.

higher latencies compared to the Hybrid schedule. However, both Hybrid and L-Hybrid schedules provide lower latency for all links compared to the other two schedules, under all data generation patterns. This is because each node can use the first unused hybrid timeslot to deliver its packet and reach low latencies, while at the worst case if there is no free non-owner hybrid timeslot available, it uses the hybrid timeslot that it owns. The Dedicated schedule performs well for all the links under the periodic scenario, in which the packet generation period is longer than the slotframe size and packets are not queued in the MAC buffers. For the Periodic-dyn. and Event-based scenarios in which the application data generation rate may go temporarily higher than the supported bandwidth by the dedicated schedule, this schedule shows higher data delivery latencies due to packet buffering. Because of the contention-based communications in the Shared schedule, this schedule performs poor under all scenarios, except in the Het.-low rate scenario in which it performs better than the Dedicated schedule. This is because when the application data rate is low, the contention on accessing a shared timeslot is also low. Thus, with a high probability, a packet can be successfully transmitted on the first shared timeslot right after its generation. On the other hand, high data generation rates lead to more contention and use of long back-off windows, causing long latencies for the Shared schedule. This shows that shared timeslots are more suitable for low data rates.

Figure 6.3 shows the distribution of all links' average PRR for different schedules and scenarios. The Dedicated, Hybrid, and L-Hybrid schedules can handle the data traffic and deliver almost all packets. This is because all these schedules dedicate an amount of bandwidth to each link and guarantee transmission of a basic data rate for the application. However, the Shared schedule cannot guarantee a bandwidth for each link and the provided bandwidth is highly depending on the data rate of other links. Therefore, the Shared schedule only provides good communication reliability when data generation rate is low (e.g., Het.-low rate scenario).

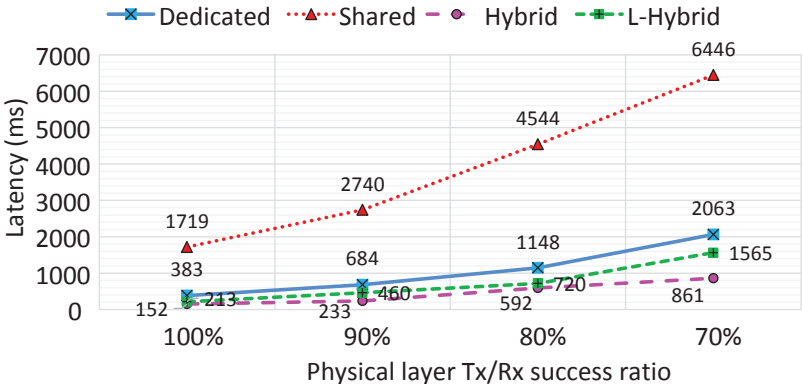


Figure 6.4: Average end-to-end latency (packet generation to receiving acknowledgment) for heterogeneous data generation scenario under different physical layer Tx/Rx success ratios.

In general, our lab experiments prove the functionality of the hybrid timeslots as a new type of timeslot for the TSCH protocol. The results show the positive effect of this new type of timeslot on reducing the end-to-end communication latency in a TSCH network. Moreover, this latency reduction does not affect reliability of the communications.

6.4.3 Simulation Results

We use the same source code that is used for our lab experiments to perform simulations in the COOJA simulator. Here we investigate the performance of different schedules under different physical layer reliability levels (Tx/Rx success ratios) for the Het.-high rate scenario.

Figure 6.4 shows the average link latency for different schedules. The Hybrid schedule reduces the latency about half the average latency of the Dedicated schedule, as every node can use the first free hybrid timeslot for packet transmission. While the same back-off mechanism is used in the hybrid and shared timeslots for non-owner access, the average latency of the Hybrid schedule is about one tenth of the latency of the Shared schedule. This is because, in the worst case, data of a node is transmitted in the dedicated timeslot to that node with a latency equal to the latency of the Dedicated schedule. However, if there is a timeslot closer to the packet generation time of which the owner skips transmission, the node has the chance to transmit the packet, leading to a lower average latency.

Since most of the traffic in the Hybrid schedule is transferred in owner timeslots, only a part of the traffic is transmitted in the non-owner timeslots. This leads to less contention on accessing non-owner timeslots and use of shorter back-off windows compared to the Shared schedule, leading to lower latencies for Hybrid schedule. Furthermore, due to the heterogeneity of the data generation, different nodes may have different data rates at any point of time. Using hybrid timeslots, a node that has a high

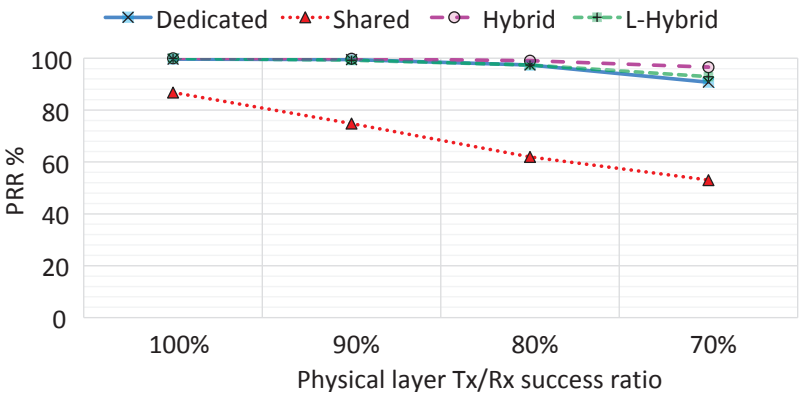


Figure 6.5: Average application PRR for heterogeneous data generation under different physical layer Tx/Rx success ratios.

data generation rate in a period of time can steal the bandwidth allocated to another node with a lower data rate in that period. This leads to less packet queuing of Hybrid and L-Hybrid schedules compared to the Dedicated schedule and lower communication latency for them. As the L-Hybrid schedule uses longer timeslots, it provides higher communication latencies compared to the Hybrid schedule. For the same reason, the L-Hybrid schedule reduces the TSCH MAC bandwidth that results in more contention on the access to non-owner timeslots, which again increases the average communication latency. However, the provided average communication latency by the L-Hybrid schedule is still much lower compared to the Dedicated and Shared schedules.

Figure 6.5 shows that using both Hybrid and L-Hybrid schedules provides higher PRR compared to the Shared schedule, for different physical layer transmission success ratios. This is because each link in a Hybrid schedule is the owner of one timeslot and has a minimum guaranteed bandwidth (minimum PRR is equal to the Dedicated schedule), while in a Shared schedule, no bandwidth is guaranteed meaning that the contention probability determines the PRR. More contention on accessing shared timeslots leads to more retransmissions and more waiting packets in the MAC buffer that causes buffer overflow and packet drops. For lower physical layer transmission success ratios, both schedules with hybrid timeslots even performs better than the dedicated schedule. This is because when nodes experience more transmission failures, they need to keep the packet in the MAC buffer and retransmit it for the maximum retransmission count. For the Dedicated schedule, each retransmission leads to a delay of one slotframe for all the packets in the buffer. As this buffer has a limited size, it may get full some times and newly generated packets can be dropped. However, a schedule with hybrid timeslots shares the unused allocated bandwidth to a node with other nodes that may have packets waiting in the buffer. This reduces the probability of packet drops that may be caused by MAC buffer overflow.

We also investigate the average number of transmissions at the MAC layer to successfully deliver a packet. This metric gives an estimation of the average power that is consumed to deliver one packet on a link. Figure 6.6 shows the results for differ-

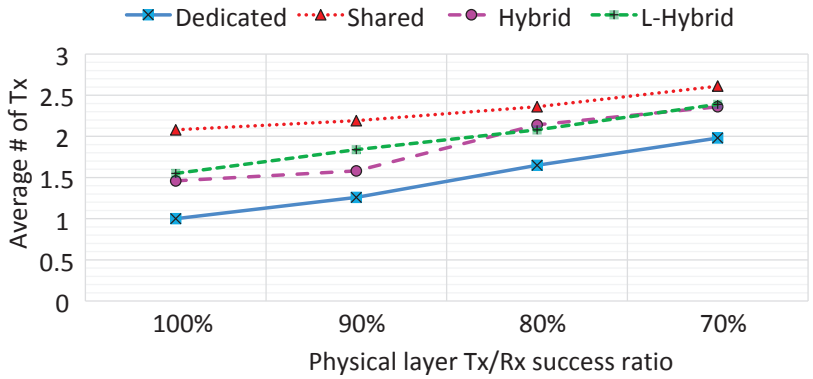


Figure 6.6: Average number of transmissions for heterogeneous data generation under different physical layer Tx/Rx success ratios.

ent techniques and physical layer transmission success ratios. When the physical links are 100% reliable, as there is no disturbance for the dedicated communications, all packets can be delivered by only one transmission. However, for the Shared schedule that uses contention-based communications in all timeslots, on average, more than one transmission is needed to successfully deliver each packet. As hybrid timeslots inherit the specifications of both dedicated and shared timeslots. and a node may use these timeslots either as dedicated or shared, the average number of transmissions of hybrid timeslots sits between those of the Dedicated and Shared schedules. The retransmission cost is actually for reaching lower communication latencies (in line with observations made in [84]). However, the average transmission count of the Hybrid and L-Hybrid schedules is increasing with the same slope as for the Dedicated schedule, when physical layer transmission success ratio decreases. This shows that the ratio between power consumption of these two schedules is getting closer for lower communication reliabilities. For the Shared schedule, although less packets are successfully delivered for lower physical layer reliability, the average number of transmissions is also increasing for the delivered packets. This shows that under high data generation rates, the Shared schedule performs poor in term of power consumption, as well as reliability and latency. However, both hybrid schedules provide better reliability and latency compared to the Dedicated schedule, at the cost of more power consumption.

6.5 Summary

This chapter presented a new type of timeslots for the IEEE 802.15.4 TSCH protocol, called hybrid timeslots. Hybrid timeslots are proposed to support heterogeneity and time-varying behavior in the data generation in automotive WSNs. Each hybrid timeslot has an owner transmitter that uses the timeslot as a normal dedicated timeslot. If the owner skips transmission in a hybrid timeslot, that timeslot can be used as a shared timeslot by all the other nodes that have a packet towards the same destination as the owner transmitter. Experimental and simulation results show that using hybrid timeslots

instead of dedicated timeslots in a TSCH schedule reduces the average communication latency by half, when maximum retransmission count is fixed. Moreover, this technique improves the reliability of communications by reducing MAC failures due to buffer overflow. The technique comes with a small increase in power consumption that is still lower than the power consumption of a schedule with only shared timeslots. This technique also fits well with the LLTT technique proposed in Chapter 5. Using hybrid timeslots, grouped retransmission timeslots can be skipped in LLTT, which leads to shorter slotframes and reduced communication latency for automotive WSNs.

7

Conclusions and Future Work

Using Wireless Sensor Networks (WSNs) for in-vehicle communications reduces wiring and manufacturing costs, and increases the flexibility and re-configurability of the in-vehicle networks. This has attracted a lot of attention from both industry and academia to develop dependable wireless communication solutions for automotive applications. Employing general-purpose solutions for these networks may not meet the stringent quality of service requirements of the automotive applications. Unique characteristics of automotive WSNs make them different from typical WSNs. This includes small area, high node density, sensor and actuator variations, and temporal variations that are partly caused by the network operation mode and partly by the wireless medium dynamics. To provide a dependable protocol stack for automotive WSNs, these characteristics should be considered for the network design.

Existing state-of-the-art solutions focus on the TSCH mode of the IEEE 802.15.4 [7] standard as the MAC layer for these networks. TSCH provides guaranteed access to the medium for network connections, which makes it a good option for automotive applications with stringent requirements. Moreover, the time sharing nature of the TSCH mechanism prevents intra-network collisions caused by high node density. TSCH also employs a channel hopping technique, which mitigates blocking of wireless links due to interference and multipath fading. The IEEE 802.15.4 protocol and TSCH are selected as the physical and MAC layers of our protocol stack for automotive WSNs. Accordingly, in this thesis, different techniques are proposed on different layers of the networking protocol stack, to address dependability issues of automotive WSNs. In the remainder of this chapter, Section 7.1 summarizes the contributions of this thesis and Section 7.2 gives some recommendations for the future work directions.

7.1 Conclusions

IEEE 802.15.4 WSNs are expected to be affected considerably by the other coexisting technologies. Accordingly, knowing the cross-technology interference behavior is a prerequisite to develop techniques that mitigate the effect of this interference. This is more challenging for automotive WSNs with continuously changing environments. In

Chapter 3, real-world experiments with different scenarios are described to study the cross-technology interference in the automotive environments. The measurement results show that interference affects all of the IEEE 802.15.4 channels, but distribution of the interference power is not uniform over time and channels. However, the interference power on each channel is often stable over substantial periods of time. The measurement data set was used as an input for simulations to study performance of the TSCH protocol. The results show that interference from in-car sources leads to considerable packet errors that are almost uniform over time. On the other hand, for out-of-car interference sources, the probability of packet errors can be highly dynamic over time. These observations show the necessity of using adaptive techniques to mitigate the dynamic interference in automotive environments.

According to the observations of Chapter 3, Chapter 4 proposes Enhanced Timeslotted Channel Hopping with Distributed Channel Sensing (ETSCH+DCS), an adaptive interference mitigation technique on top of the IEEE 802.15.4 [7] TSCH protocol. This mechanism adaptively extracts the quality of different wireless channels over time to select the channels with the lowest interference as the hopping sequence list to improve the performance of the TSCH protocol. Part of this technique is central and operates at the coordinator of the network and proactively measures the interference power on each channel. The other part of this technique is distributed and is performed by all the nodes in the network and uses the CCA results together with packet reception status to estimate the interference power on each channel. The results of the centralized and distributed channel quality estimation techniques are used to assign quality grades to each wireless channel. Then, channels with better qualities are periodically selected as the hopping sequence list of TSCH. Experimental and simulation results show that this technique provides higher reliability and lower length of burst packet losses, compared to the plain TSCH protocol and another related work called ATSCH. However, simulation results based on the real-world data sets (presented in Chapter 3) show that ETSCH is not able to completely mitigate the effect of interference. Simulations also show that by employing two retransmissions at the MAC layer, ETSCH provides a link-level reliability greater than 95%. It should be considered that retransmissions increase the average latency of communications which may lead to invalidation of the data. Accordingly, the TSCH schedule should be managed carefully to enable more potential retransmissions within the defined latency bounds.

Chapter 5 presents Low-Latency Topology management and TSCH scheduling (LLTT), a cross-layer design for reliable and low-latency data convergecast in dense TSCH-based networks. This technique reduces the latency of end-to-end communications by managing the network topology and the TSCH schedule. It firstly picks a tree-topology structure with maximum number of independent links for the network. This increases the potential of parallelism in the TSCH schedule. Employing an optimized graph isomorphism algorithm, LLTT extracts a proper match for the selected tree structure in the connectivity graph of the network. The extracted network topology is used by a light-weight central TSCH schedule generator to find a highly parallelized TSCH schedule with dedicated timeslots to links. This allocates a constant bandwidth to each link, as timeslots repeat over time with a periodic pattern. This schedule generator allocates grouped retransmission timeslots among dedicated timeslots to improve commu-

nication reliability. Periodic data aggregation is used to improve bandwidth utilization. Experimental results in an in-vehicle testbed with 31 sensor nodes and in presence of interference generators show that LLTT is able to provide latencies below 250 ms, while it keeps communications reliable by using retransmissions.

Due to the spatial and temporal heterogeneity of automotive WSNs, the required bandwidth by each link may change over time. This leads to waste or shortage of communication resources over different periods of time. Accordingly, Chapter 6 presents hybrid timeslots to support heterogeneity and time-varying behavior in the data generation in automotive WSNs. Each hybrid timeslot has an owner transmitter that uses the timeslot as a normal dedicated timeslot. If the owner skips transmission in a hybrid timeslot, that timeslot can be used as a shared timeslot by all the other nodes that have a packet towards the same destination as the owner transmitter. This reduces the need for extra dedicated or shared timeslots in a TSCH schedule for retransmissions. Experimental and simulation results show that using hybrid timeslots instead of dedicated timeslots in a TSCH schedule reduces the average communication latency substantially. Moreover, this technique improves the reliability of communications by reducing MAC failures due to buffer overflow. The technique comes with a small increase in power consumption that is still lower than the power consumption of a schedule with only shared timeslots.

All the proposed techniques in this thesis, namely ETSCH+DCS, LLTT, and hybrid timeslots, can be used together in the automotive WSN protocol stack. ETSCH+DCS picks less interfered channels for channel hopping to increase reliability, LLTT provides a short slotframe to reduce latency, and hybrid timeslots support heterogeneity and time-varying behavior of data traffic and improves both reliability and latency. Moreover, by using hybrid timeslots, grouped retransmission timeslots can be skipped in LLTT. This leads to shorter slotframes and reduced communication latency for automotive WSNs. It should be considered that transmission of dummy packets in unused dedicated timeslots, that is used by DCS technique, is not applicable for hybrid timeslots.

7.2 Future Work

Although great research has been done in the field of automotive WSNs, there is still a wide range of open research questions to be answered in this field. This section highlights the opportunities to extend the work of this thesis and improve the performance of automotive WSNs.

- To get the best performance of a TSCH WSN, it is very important to configure it properly based on the application behavior and requirements. This requires extensive experiments with real-world setups or simulations to extract the performance of the network for different configurations. This is usually costly in terms of time, if different combinations of various parameters are considered. High level analytical modeling of WSNs is also one of the performance evaluation options. General analytical models such as Markov model can be very complex and may even cost more time. However, specialized analytical models that are optimized for the case under study can provide accurate results in very short times. Accordingly, analyt-

ical models for performance evaluation of TSCH-based WSNs is a future work direction. To the best of our knowledge, there is currently no analytical performance model for TSCH networks that takes the TSCH schedule and physical properties of the network into account. This kind of analytical model can be used to extract the number of required dedicated and shared timeslots in a TSCH schedule to provide an expected QoS level for the application.

- Scalability of WSNs is one of the important aspects that should be considered to design most of the WSNs. However, scalability is not a requirement for the automotive applications; thus we considered a non-scalable network in this thesis. Accordingly, to reuse some of the proposed mechanisms in this work for other applications, some modifications are required. ETSCH is one of the techniques that can be used for a wide range of applications, in which interference from coexisting devices affects the reliability of communications. However, ETSCH is a central technique which suits small size networks and needs some modifications to be used for scalable networks. A possible strategy could be to use multiple gateways (coordinators) in the network to build multiple non-overlapping clusters. Each gateway can perform ETSCH to detect the less interfered channels in the range of its cluster and use them as the hopping sequence list of the nodes in that cluster. However, if neighbor clusters use different hopping sequence lists for their communications, they may cause interference for each other. To avoid this, it is needed to perform some kind of negotiation between the gateways of neighbor clusters.

- The LLTT technique that was proposed in Chapter 5 is mainly designed to be run every once in a while; e.g., when the vehicle is turned on. Accordingly, link qualities at that time are used to build the network topology. However, communication links may experience long term quality changes that may be caused by passengers or objects in the vehicle (load of the truck, for example). This requires run-time reconfiguration of the network topology, according to the environment changes. This should be done without affecting the network functionality. Replacing bad links with better ones and using adaptive transmission power are some of the possible solutions.

- Cooperative coexistence is the idea of managing different coexisting wireless technologies, which operate at the same frequency band, to provide conflict-free communications over time and/or frequency domains. This topic attracted a lot of attention in recent years [17, 18, 78]. As it is shown in Chapter 3, interference of coexisting devices in a car has a high impact on the performance of in-vehicle WSNs. Considering the embedded wireless-enabled devices in a vehicle, such as a Bluetooth audio system and on-board Wi-Fi access point, cooperative coexistence mechanisms can be developed to improve the performance of in-vehicle wireless technologies.

Bibliography

[1] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 15.1a: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPAN). *IEEE Std 802.15.1-2005 (Revision of IEEE Std 802.15.1-2002)*, pages 1–700, June 2005. (Cited on pages 7, 23, 24, and 45.)

[2] IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low Rate Wireless Personal Area Networks (WPANs). *IEEE Std 802.15.4-2006 (Revision of IEEE Std 802.15.4-2003)*, pages 1–320, Sept 2006. (Cited on pages 1, 24, 37, and 38.)

[3] *ISA, ISA100.11a:2009 Wireless systems for industrial automation: Process control and related applications, International Society of Automation Std.*, 2009. (Cited on page 12.)

[4] *ANSI/ISA-100.11a-2011 Wireless systems for industrial automation: Process control and related applications, International Society of Automation Std.*, 2011. (Cited on page 46.)

[5] 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)*, pages 1–2793, March 2012. (Cited on pages 7, 23, 24, 45, and 47.)

[6] IEEE Standard for High Data Rate Wireless Multi-Media Networks. *IEEE Std 802.15.3-2016 (Revision of IEEE Std 802.15.3-2003*, pages 1–510, July 2016. (Cited on page 24.)

[7] IEEE Standard for Low-Rate Wireless Networks. *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, pages 1–709, April 2016. (Cited on pages 4, 6, 12, 13, 52, 56, 58, 59, 61, 78, 83, 103, 105, 106, 107, 117, and 118.)

[8] Nicola Accettura, Maria Rita Palattella, Gennaro Boggia, Luigi Alfredo Grieco, and Mischa Dohler. Decentralized traffic aware scheduling for multi-hop low power lossy networks in the internet of things. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2013 IEEE 14th International Symposium and Workshops on a*, pages 1–6. IEEE, 2013. (Cited on pages 15, 81, and 84.)

[9] Kaoru Amano, Naokazu Goda, Shin'ya Nishida, Yoshimichi Ejima, Tsunehiro Takeda, and Yoshio Ohtani. Estimation of the timing of human visual perception from magnetoencephalography. *Journal of Neuroscience*, 26(15):3981–3991, 2006. (Cited on page 5.)

[10] L. Angrisani, M. Bertocco, D. Fortin, and A. Sona. Experimental Study of Coexistence Issues Between IEEE 802.11b and IEEE 802.15.4 Wireless Networks. *IEEE Transactions on Instrumentation and Measurement*, 57(8):1514–1523, Aug 2008. (Cited on page 24.)

[11] Algirdas Avizienis, J-C Laprie, Brian Randell, and Carl Landwehr. Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1):11–33, Jan 2004. (Cited on page 4.)

[12] Nouha Baccour, Anis Koubâa, Luca Mottola, Marco Antonio Zúñiga, Habib Youssef, Carlo Alberto Boano, and Mário Alves. Radio Link Quality Estimation in Wireless Sensor Networks: A Survey. *ACM Trans. Sen. Netw.*, 8(4):34:1–34:33, September 2012. (Cited on page 24.)

[13] Miloud Bagaa, Mohamed Younis, Djamel Djenouri, Abdelouahid Derhab, and Nadjib Badache. Distributed Low-Latency Data Aggregation Scheduling in Wireless Sensor Networks. *ACM Transaction on Sensor Networks.*, 11(3):49:1–49:36, April 2015. (Cited on pages 15, 81, and 84.)

[14] C. U. Bas and S. C. Ergen. Ultra-wideband channel model for intra-vehicular wireless sensor networks beneath the chassis: From statistical model to simulations. *IEEE Transactions on Vehicular Technology*, 62(1):14–25, Jan 2013. (Cited on page 37.)

[15] D. Block, N. H. Fliedner, D. Toews, and U. Meier. Wireless channel measurement data sets for reproducible performance evaluation in industrial environments. In *2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–4, Sept 2015. (Cited on page 25.)

[16] Bluetooth SIG, Inc. Bluetooth Specification Version 5.0. `https://www.bluetooth.com/specifications/bluetooth-core-specification`. (Cited on pages 23 and 24.)

[17] O. Carhacioglu, P. Zand, and M. Nabi. Time-domain cooperative coexistence of BLE and IEEE 802.15.4 networks. In *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–7, Oct 2017. (Cited on page 120.)

[18] Onur Carhacioglu, Pouria Zand, and Majid Nabi. Cooperative Coexistence of BLE and Time Slotted Channel Hopping Networks. In *2018 IEEE 29th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Bologna, Italy, September 2018. (Cited on page 120.)

[19] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento. A (sub)graph isomorphism algorithm for matching large graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1367–1372, Oct 2004. (Cited on page 87.)

[20] R. de Francisco, Li Huang, G. Dolmans, and H. de Groot. Coexistence of ZigBee wireless sensor networks and Bluetooth inside a vehicle. In *2009 IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications*, pages 2700–2704, Sept 2009. (Cited on page 25.)

[21] U. Demir, C. U. Bas, and S. Coleri Ergen. Engine compartment uwb channel model for intravehicular wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 63(6):2497–2505, July 2014. (Cited on page 37.)

[22] Sadia Din, Anand Paul, Awais Ahmad, and Jeong Hong Kim. Energy efficient topology management scheme based on clustering technique for software defined wireless sensor network. *Peer-to-Peer Networking and Applications*, Oct 2017. (Cited on page 84.)

[23] P. Du and G. Roussos. Adaptive time slotted channel hopping for wireless sensor networks. In *2012 4th Computer Science and Electronic Engineering Conference (CEEC)*, pages 29–34, Sept 2012. (Cited on pages 48, 56, and 64.)

[24] Adam Dunkels, Bjorn Gronvall, and Thiemo Voigt. Contiki-a lightweight and flexible operating system for tiny networked sensors. In *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pages 455–462. IEEE, 2004. (Cited on pages 21 and 109.)

[25] Simon Duquennoy, Beshr Al Nahas, Olaf Landsiedel, and Thomas Watteyne. Orchestra: Robust Mesh Networks Through Autonomously Scheduled TSCH. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, SenSys '15, pages 337–350, New York, NY, USA, 2015. ACM. (Cited on pages 15, 81, 84, and 96.)

[26] A. Elsts, X. Fafoutis, R. Piechocki, and I. Craddock. Adaptive channel selection in ieee 802.15.4 tsch networks. In *2017 Global Internet of Things Summit (GIoTS)*, pages 1–6, June 2017. (Cited on page 48.)

[27] A. Elsts, X. Fafoutis, J. Pope, G. Oikonomou, R. Piechocki, and I. Craddock. Scheduling High-Rate Unpredictable Traffic in IEEE 802.15.4 TSCH Networks. In *2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pages 3–10, June 2017. (Cited on pages 104 and 105.)

[28] David Eppstein. Subgraph isomorphism in planar graphs and related problems. In *SODA*, volume 95, pages 632–640, 1995. (Cited on pages 82 and 87.)

[29] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi. In-network aggregation techniques for wireless sensor networks: a survey. *IEEE Wireless Communications*, 14(2):70–87, April 2007. (Cited on page 84.)

[30] G. Gaillard, D. Barthel, F. Theoleyre, and F. Valois. High-reliability scheduling in deterministic wireless multi-hop networks. In *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–6, Sept 2016. (Cited on page 84.)

[31] Shashidhar Gandham, Milind Dawande, and Ravi Prakash. Link scheduling in wireless sensor networks: distributed edge-coloring revisited. *Journal of Parallel and Distributed Computing*, 68(8):1122–1134, 2008. (Cited on page 83.)

[32] Everette S. Gardner and David G. Dannenbring. FORECASTING WITH EXPONENTIAL SMOOTHING: SOME GUIDELINES FOR MODEL SELECTION. *Decision Sciences*, 11(2):370–383, 1980. (Cited on page 55.)

[33] J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, 41(2):148–177, 1979. (Cited on page 47.)

[34] Pedro Henrique Gomes, Thomas Watteyne, and Bhaskar Krishnamachari. Mabotsch: Multihop and blacklist-based optimized time synchronized channel hopping. *Trans. Emerg. Telecommun. Technol.*, pages e3223–n/a, 2017. (Cited on page 47.)

[35] Kees Goossens, Martijn Koedam, Andrew Nelson, Shubhendu Sinha, Sven Goossens, Yonghui Li, Gabriela Breaban, Reinier van Kampenhout, Rasool Tavakoli, Juan Valencia, Hadi Ahmadi Balef, Benny Akesson, Sander Stuijk, Marc Geilen, Dip Goswami, and Majid Nabi. *NoC-Based Multiprocessor Architecture for Mixed-Time-Criticality Applications*, pages 491–530. Springer Netherlands, Dordrecht, 2017. (Cited on page 137.)

[36] V. C. Gungor and G. P. Hancke. Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches. *IEEE Transactions on Industrial Electronics*, 56(10):4258–4265, Oct 2009. (Cited on page 18.)

[37] W. Guo, W. M. Healy, and M. Zhou. Impacts of 2.4-GHz ISM Band Interference on IEEE 802.15.4 Wireless Sensor Network Reliability in Buildings. *IEEE Transactions on Instrumentation and Measurement*, 61(9):2533–2544, Sept 2012. (Cited on pages 24 and 25.)

[38] S. A. Hanna and J. Sydor. Distributed sensing of spectrum occupancy and interference in outdoor 2.4 GHz Wi-Fi networks. In *2012 IEEE Global Communications Conference (GLOBECOM)*, pages 1453–1459, Dec 2012. (Cited on page 25.)

[39] M. Hashemi, W. Si, M. Laifenfeld, D. Starobinski, and A. Trachtenberg. Intra-car multihop wireless sensor networking: a case study. volume 52, pages 183–191, December 2014. (Cited on page 83.)

[40] M. Hashimoto, N. Wakamiya, M. Murata, Y. Kawamoto, and K. Fukui. End-to-end reliability- and delay-aware scheduling with slot sharing for wireless sensor networks. In *8th International Conference on Communication Systems and Networks (COMSNETS)*, pages 1–8, 2016. (Cited on pages 84, 104, and 105.)

[41] Jan-Hinrich Hauer, Vlado Handziski, and Adam Wolisz. Experimental Study of the Impact of WLAN Interference on IEEE 802.15.4 Body Area Networks. In Utz Roedig and Cormac J. Sreenan, editors, *Wireless Sensor Networks*, pages 17–32. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. (Cited on pages 24 and 25.)

[42] Ta-Yang Huang, Chia-Jui Chang, C. W. Lin, Sudip Roy, and Tsung-Yi Ho. Intra-vehicle network routing algorithm for wiring weight and wireless transmit power minimization. In *The 20th Asia and South Pacific Design Automation Conference*, pages 273–278, Jan 2015. (Cited on page 84.)

[43] K. Jeon and S. Chung. Adaptive channel quality estimation method for enhanced time slotted channel hopping on wireless sensor networks. In *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 438–443, July 2017. (Cited on page 49.)

[44] Y. Jin, P. Kulkarni, J. Wilcox, and M. Sooriyabandara. A centralized scheduling algorithm for IEEE 802.15.4e TSCH based industrial low power wireless networks. In *2016 IEEE Wireless Communications and Networking Conference*, pages 1–6, April 2016. (Cited on pages 15, 81, and 84.)

[45] Holger Karl and Andreas Willig. *Protocols and architectures for wireless sensor networks*. John Wiley & Sons, 2007. (Cited on page 37.)

[46] Peishuo Li, T. Vermeulen, H. Liy, and S. Pollin. An adaptive channel selection scheme for reliable tsch-based communication. In *2015 International Symposium on Wireless Communication Systems (ISWCS)*, pages 511–515, Aug 2015. (Cited on pages 47 and 49.)

[47] L. LoBello and E. Toscano. An adaptive approach to topology management in large and dense real-time wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 5(3):314–324, Aug 2009. (Cited on page 84.)

[48] J. Luo, J. Hu, D. Wu, and R. Li. Opportunistic routing algorithm for relay node selection in wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 11(1):112–121, Feb 2015. (Cited on page 84.)

[49] MEMSIC Inc. TelosB mote platform. `http://www.memsic.com/userfiles/files/Datasheets/WSN/telosb_datasheet.pdf`. Accessed: March. 2018. (Cited on page 109.)

[50] Microchip Technology Inc. ATmega256RFR2 Xplained Pro Evaluation Kit. `http://www.microchip.com/`. (Cited on pages 19, 21, 58, and 63.)

[51] R. Natarajan, P. Zand, and M. Nabi. Analysis of coexistence between ieee 802.15.4, ble and ieee 802.11 in the 2.4 ghz ism band. In *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, pages 6025–6032, Oct 2016. (Cited on page 24.)

[52] E. D. Ngangue Ndih, S. Cherkaoui, and I. Dayoub. Analytic Modeling of the Coexistence of IEEE 802.15.4 and IEEE 802.11 in Saturation Conditions. *IEEE Communications Letters*, 19(11):1981–1984, Nov 2015. (Cited on page 24.)

[53] NXP Semiconductors. JN516x IEEE802.15.4 Wireless Microcontroller. `http://www.nxp.com/docs/en/data-sheet/JN516X.pdf`. Accessed: Aug. 2018. (Cited on pages 19, 21, 95, and 109.)

[54] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt. Cross-Level Sensor Network Simulation with COOJA. In *Proceedings. 31st IEEE Conference on Local Computer Networks*, pages 641–648, Nov 2006. (Cited on pages 21, 22, and 109.)

[55] M. R. Palattella, T. Watteyne, Q. Wang, K. Muraoka, N. Accettura, D. Dujovne, L. A. Grieco, and T. Engel. On-the-Fly Bandwidth Reservation for 6TiSCH Wireless Industrial Networks. *IEEE Sensors Journal*, 16(2):550–560, Jan 2016. (Cited on pages 15, 81, 104, and 105.)

[56] Maria Rita Palattella, Nicola Accettura, Mischa Dohler, Luigi Alfredo Grieco, and Gennaro Boggia. Traffic Aware Scheduling Algorithm for reliable low-power multihop IEEE 802.15. 4e networks. In *Personal Indoor and Mobile Radio Communications (PIMRC), 2012 IEEE 23rd International Symposium on*, pages 327–332. IEEE, 2012. (Cited on pages 15, 81, and 84.)

[57] D. Parthasarathy, E. Wermström, and R. Whiton. Autmotive architecture and integration requirements. Technical Report D308.001, European project DEWI: Dependable Embedded Wireless Infrastructure, September 2014. (Cited on pages 2 and 83.)

[58] D. Parthasarathy, R. Whiton, J. Hagerskans, and T. Gustafsson. An in-vehicle wireless sensor network for heavy vehicles. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8, Sept 2016. (Cited on page 83.)

[59] F. Penna, C. Pastrone, M. A. Spirito, and R. Garello. Measurement-Based Analysis of Spectrum Sensing in Adaptive WSNs under Wi-Fi and Bluetooth Interference. In *VTC Spring 2009 - IEEE 69th Vehicular Technology Conference*, pages 1–5, April 2009. (Cited on page 24.)

[60] M. Petrova, J. Riihijarvi, P. Mahonen, and S. Labella. Performance study of ieee 802.15.4 using measurements and simulations. In *IEEE Wireless Communications and Networking Conference, 2006. WCNC 2006.*, volume 1, pages 487–492, April 2006. (Cited on page 24.)

[61] Md. Arafatur Rahman. Design of wireless sensor network for intra-vehicular communications. In Abdelhamid Mellouk, Scott Fowler, Saïd Hoceini, and Boubaker Daachi, editors, *Wired/Wireless Internet Communications*, pages 29–40, Cham, 2014. Springer International Publishing. (Cited on page 83.)

[62] Rasool Tavakoli. In-Vehicle Interference Data Set. `https://git.ics.ele.tue.nl/Public0/interference-behavior-in-in-vehicle-env`, 2018. (Cited on pages 36 and 42.)

[63] Stefan Reis, Dirk Pesch, Bernd-Ludwig Wenning, and Michael Kuhn. Intra-vehicle wireless sensor network communication quality assessment via packet delivery ratio measurements. In Ramón Agüero, Yasir Zaki, Bernd-Ludwig Wenning, Anna Förster, and Andreas Timm-Giel, editors, *Mobile Networks and Management*, pages 88–101, Cham, 2017. Springer International Publishing. (Cited on page 83.)

[64] Injong Rhee, Ajit Warrier, Mahesh Aia, Jeongki Min, and Mihail L. Sichitiu. Z-MAC: A Hybrid MAC for Wireless Sensor Networks. *IEEE/ACM Trans. Netw.*, 16(3):511–524, June 2008. (Cited on page 105.)

[65] M. Schack, J. Jemai, R. Piesiewicz, R. Geise, I. Schmidt, and T. Kurner. Measurements and analysis of an in-car uwb channel. In *VTC Spring 2008 - IEEE Vehicular Technology Conference*, pages 459–463, May 2008. (Cited on page 37.)

[66] C. F. Shih, A. E. Xhafa, and J. Zhou. Practical frequency hopping sequence design for interference avoidance in 802.15.4e tsch networks. In *2015 IEEE International Conference on Communications (ICC)*, pages 6494–6499, June 2015. (Cited on page 61.)

[67] Soo Young Shin. Throughput analysis of IEEE 802.15. 4 network under IEEE 802.11 network interference. *AEU-International Journal of Electronics and Communications*, 67(8):686–689, 2013. (Cited on page 24.)

[68] Jianping Song, Song Han, Al Mok, Deji Chen, Mike Lucas, Mark Nixon, and Wally Pratt. WirelessHART: Applying wireless technology in real-time industrial process control. In *Real-Time and Embedded Technology and Applications Symposium, 2008. RTAS'08. IEEE*, pages 377–386. IEEE, 2008. (Cited on pages 12 and 46.)

[69] Ridha Soua, Pascale Minet, and Erwan Livolant. Wave: a distributed scheduling algorithm for convergecast in IEEE 802.15. 4e TSCH networks. *Transactions on Emerging Telecommunications Technologies*, 27(4):557–575, 2016. (Cited on page 84.)

[70] James Andrew Storer. *An introduction to data structures and algorithms*. Springer Science & Business Media, 2012. (Cited on page 87.)

[71] Rasool Tavakoli and Majid Nabi. TIGeR: A Traffic-Aware Intersection-Based Geographical Routing Protocol for Urban VANETs. In *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, pages 1–5, June 2013. (Cited on page 137.)

[72] Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Enhanced Time-Slotted Channel Hopping in WSNs Using Non-intrusive Channel-Quality Estimation. In *Mobile Ad Hoc and Sensor Systems (MASS), 2015 IEEE 12th International Conference on*, pages 217–225, Oct 2015. (Cited on pages 9, 46, and 137.)

[73] Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. An Experimental Study of Cross-Technology Interference in In-Vehicle Wireless Sensor Networks. In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, pages 195–204, New York, NY, USA, 2016. ACM. (Cited on pages 9, 24, and 137.)

[74] Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Dependable Interference-Aware Time-Slotted Channel Hopping for Wireless Sensor Networks. *ACM Transactions on Sensor Networks*, 14(1):3:1–3:35, January 2018. (Cited on pages 9, 46, and 137.)

[75] Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Guard-Time Design for Symmetric Synchronization in IEEE 802.15.4 Time-Slotted Channel Hopping. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, June 2018. (Cited on pages 107 and 137.)

[76] Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Hybrid Timeslot Design for IEEE 802.15.4 TSCH to Support Heterogeneous WSNs. In *2018 IEEE 29th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Bologna, Italy, September 2018. (Cited on pages 9, 104, and 137.)

[77] Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Topology Management and TSCH Scheduling for Low-Latency Convergecast in In-Vehicle WSNs. *IEEE Transactions on Industrial Informatics*, 2018. (Cited on pages 9, 83, and 137.)

[78] Rambabu A. Vatti and Arun N. Gaikwad. Frame Converter for Cooperative Coexistence Between IEEE 802.15.4 Wireless Sensor Networks and Wi-Fi. In Atulya Nagar, Durga Prasad Mohapatra, and Nabendu Chaki, editors, *Proceedings of 3rd International Conference on Advanced Computing, Networking and Informatics*, pages 151–157, New Delhi, 2016. Springer India. (Cited on page 120.)

[79] R. Velmani and B. Kaarthick. An efficient cluster-tree based data collection scheme for large mobile wireless sensor networks. *IEEE Sensors Journal*, 15(4):2377–2390, April 2015. (Cited on page 84.)

[80] X Vilajosana and K Pister. Minimal 6TiSCH Configuration draft-ietf-6tisch-minimal-06 (work in progress). Technical report, IETF, Internet Draft, Jan. 2015.[Online]. Available: http://tools.ietf.org/html/draft-ietf-6tisch-minimal-06, 2015. (Cited on pages 83 and 96.)

[81] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. J. Pister. A realistic energy consumption model for tsch networks. *IEEE Sensors Journal*, 14(2):482–489, Feb 2014. (Cited on page 58.)

[82] Chundong Wang, Zhentang Zhao, Likun Zhu, and Honglei Yao. An energy efficient routing protocol for in-vehicle wireless sensor networks. In *Data Science*, pages 161–170, Singapore, 2017. Springer Singapore. (Cited on page 84.)

[83] Qin Wang, Xavier Vilajosana, and Thomas Watteyne. 6top Protocol (6P). Internet-Draft draft-ietf-6tisch-6top-protocol-07, Internet Engineering Task Force, June 2017. Work in Progress. (Cited on page 105.)

[84] T. Watteyne, J. Weiss, L. Doherty, and J. Simon. Industrial IEEE802.15.4e networks: Performance and trade-offs. In *2015 IEEE International Conference on Communications (ICC)*, pages 604–609, June 2015. (Cited on page 114.)

[85] Thomas Watteyne, Ankur Mehta, and Kris Pister. Reliability Through Frequency Diversity: Why Channel Hopping Makes Sense. In *Proceedings of the 6th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, PE-WASUN '09, pages 116–123, New York, NY, USA, 2009. ACM. (Cited on pages 45, 47, and 62.)

[86] Hong wei HUO, You zhi XU, Gidlund Mikael, and Hong ke ZHANG. Coexistence of 2.4 GHz sensor networks in home environment. *The Journal of China Universities of Posts and Telecommunications*, 17(1):9 – 18, 2010. (Cited on page 25.)

[87] WiFi Alliance. Wi-Fi peer-to-peer (P2P) technical specification v1.7. `https://www.wi-fi.org/discover-wi-fi/specifications`, 2018. (Cited on page 32.)

[88] Tim Winter and RPL Author Team. RPL: IPv6 routing protocol for low-power and lossy networks. 2012. (Cited on page 96.)

[89] Mohamed Younis, Izzet F. Senturk, Kemal Akkaya, Sookyoung Lee, and Fatih Senel. Topology management techniques for tolerating node failures in wireless sensor networks: A survey. *Computer Networks*, 58:254 – 283, 2014. (Cited on page 84.)

[90] Wei Yuan, Xiangyu Wang, Jean-Paul M. G. Linnartz, and Ignas G. M. M. Niemegeers. Coexistence Performance of IEEE 802.15.4 Wireless Sensor Networks Under IEEE 802.11b/g Interference. *Wireless Personal Communications*, 68(2):281–302, Jan 2013. (Cited on page 24.)

[91] S. Zacharias, T. Newe, S. O'Keeffe, and E. Lewis. Coexistence measurements and analysis of IEEE 802.15.4 with Wi-Fi and bluetooth for vehicle networks. In *2012 12th International Conference on ITS Telecommunications*, pages 785–790, Nov 2012. (Cited on page 25.)

[92] H. Zhang, P. Soldati, and M. Johansson. Performance Bounds and Latency-Optimal Scheduling for Convergecast in WirelessHART Networks. *IEEE Transactions on Wireless Communications*, 12(6):2688–2696, June 2013. (Cited on page 84.)

List of Acronyms

ACK Acknowledgement . . . 13
ASN Absolute Sequence Number . . . 15
AWGN Additive White Gaussian Noise . . . 38
BEP Bit Error probability . . . 38
BER Bit Error Rate . . . 38
BLE Bluetooth Low Energy . . . 23
CCA Clear Channel Assessment . . . 13
CQE Channel-Quality Estimation . . . 50
CSMA-CA Carrier-Sense Multiple Access with Collision Avoidance . . . 14
DCS Distributed Channel Sensing . . . 7
DDR Data Delivery Ratio . . . 17
DSSS Direct Sequence Spread Spectrum . . . 12
EBSL EB hopping Sequence List . . . 49
EB Enhanced Beacon . . . 16
ECU Electronic Control Unit . . . 3
ED Energy Detection . . . 13
ETSCH Enhanced Time-Slotted Channel Hopping . . . 7
ETX Expected Transmission count . . . 47
HSL Hopping Sequence List . . . 15
IVNs In-Vehicle Networks . . . 2
LLTT Low-Latency Topology management and TSCH scheduling . . . 8
LQI Link Quality Indicator . . . 13
LR-WPANs Low-Rate Wireless Personal Area Networks . . . 12
MAC Medium Access Control . . . 2
NG Noise Generator . . . 74
NICE Non-Intrusive Channel-quality Estimation . . . 7
O-QPSK Offset Quadrature Phase-Shift Keying . . . 12
OS Operating System . . . 21

PAN Personal Area Network 48
PDR Packet Delivery Ratio 16
PER Packet Error Rate 25
PHY Physical layer 12
PRP Packet Reception Probability 17
PRR Packet Reception Ratio 16
QoS Quality-of-Service 1
SED Silent Energy Detection 55
SNR Signal-to-Noise Ratio 17
TSCH Time-Slotted Channel Hopping 6
WCPDL Worst Case Packet Delivery Latency 94
WSNs Wireless Sensor Networks 1

Acknowledgments

Every chapter in life has to come to an end sometime, and this thesis is an end to my "PhD chapter"! It was four years filled with challenges and memories. I feel very lucky to have many supportive and encouraging people around me, at both work and my personal life. Here I express my gratitude to whom helped me to go forward and never give up on burst challenges of PhD.

In the first place, I would like to express my gratitude to my first promotor, Kees Goossens, for not only his valuable guidance and suggestions that helped me in the process of my research and improving my engineering skills, but also for giving me the opportunity to work under his supervision and be part of the CompSoC group. Likewise, I would like to express my gratitude to my second promotor, Twan Basten, for his support and constructive comments that always helped me to form new ideas and find new directions. His *out of the box thinking* always inspired me to look different to problems, towards better solutions. I feel very lucky that I have had such great promotors.

I express my best thanks to my copromotor, Majid Nabi. Our ad-hoc meetings during the past four years have been an essential help for me in all the steps of my research. Majid was not only a supervisor for me but also a model for me all the time, even before my academic studies. For me, he has been always the future to go towards and it was my best chance to work with him during my PhD.

I am thankful to Prof. Sonia Heemstra, Prof. Koen Langendoen, Prof. Johan Lukkien, Dr. Bart Vermeulen, and Prof. Jan Blom for being the members of my Doctoral Committee and for their time and effort in reviewing this manuscript.

I greatly thank our group secretary, Marja de Mol, who has always been very helpful and has made the group atmosphere even more friendly and enjoyable. I also express my gratitude to Rian van Gaalen and Margot Gordon Buteijn, who have been very supportive friends during my first months here in the Netherlands.

Working in the Electronic Systems group has been awesome and is never forgotten. It gave me the chance to have a lot of new friends, from whom I learned a lot of new things. I express my best thanks to Amir, Hadi (2), Juan, Reinier, Andrew, Victor, Manil, Francesco, Gabriela, Sven, Yonghui, Luc, Shubhendu, Martijn, Mark (3), Joost, João, Robinson, Bram, Andreia, Ali (2), Alessandro, Mohammad, Sajid, Hamide, Mojtaba, Mahsa, Amin, Paul, Cumhur, Kamlesh, shima, Shayan, Mahdi and all the people I had the pleasure to work with, for their support and friendship.

The last day of spring 2014, when Marzie and I move to the Netherlands, we were sad about being far from our families. Thanks to Majid and his family, we felt home from the moment we arrived in the Netherlands. This was a great opportunity for us to have them here as a great support and we are always thankful to them. I also express my

best gratitude to the Iranian families Baram, Farajzadeh, Ghahremani, Heidari, Haghi, Hosseini, Izadi, Mardani, Mohammadi, Naderi, Rahimzadeh, Roomi zadeh, Taheri, and Tahghighi for the great time we had with them.

I express my best gratitude beyond any expression to my parents, for all their unconditional help and supporting me throughout all these years. Thank you dad for having natured my curiosity and creativity. Thank you mom for being a lovely friend to me all the time. I also thank my lovely sisters, for the happiness they brought to our family.

My special appreciations go to my best friend, my love, and my wife, Marzie. Without her love and support, I believe I could not be where I am today. No words can express, no act of gratitude can relay, no gift can represent what your love and support meant to me.

Rasool Tavakoli Najafabadi
Eindhoven, December 2018

About the Author

Rasool Tavakoli was born in Najafabad, Iran, on April 29, 1988. He received the B.S. and M.Sc. degrees in computer engineering (hardware) from the Isfahan University of Technology, Iran in 2010 and 2013, respectively. Since June 2014, he has been a Ph.D. candidate in the Electronic Systems group at the Department of Electrical Engineering of Eindhoven University of Technology, the Netherlands, of which the results are presented in this dissertation. Rasool is currently a post-doctoral researcher, continuing his research and teaching activities in the Electronic Systems group at the Department of Electrical Engineering of the Eindhoven University of Technology. His research interests include dependable and low-power wireless sensor networks and optimization of networked embedded systems.

List of Publications

Publications covered in the thesis

- Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Hybrid Timeslot Design for IEEE 802.15.4 TSCH to Support Heterogeneous WSNs. In *2018 IEEE 29th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Bologna, Italy, September 2018
- Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Topology Management and TSCH Scheduling for Low-Latency Convergecast in In-Vehicle WSNs. *IEEE Transactions on Industrial Informatics*, 2018
- Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Dependable Interference-Aware Time-Slotted Channel Hopping for Wireless Sensor Networks. *ACM Transactions on Sensor Networks*, 14(1):3:1–3:35, January 2018
- Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. An Experimental Study of Cross-Technology Interference in In-Vehicle Wireless Sensor Networks. In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, pages 195–204, New York, NY, USA, 2016. ACM
- Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Enhanced Time-Slotted Channel Hopping in WSNs Using Non-intrusive Channel-Quality Estimation. In *Mobile Ad Hoc and Sensor Systems (MASS), 2015 IEEE 12th International Conference on*, pages 217–225, Oct 2015

Publications not covered in the thesis

- Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. Guard-Time Design for Symmetric Synchronization in IEEE 802.15.4 Time-Slotted Channel Hopping. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, June 2018
- Kees Goossens, Martijn Koedam, Andrew Nelson, Shubhendu Sinha, Sven Goossens, Yonghui Li, Gabriela Breaban, Reinier van Kampenhout, Rasool Tavakoli, Juan Valencia, Hadi Ahmadi Balef, Benny Akesson, Sander Stuijk, Marc Geilen, Dip Goswami, and Majid Nabi. *NoC-Based Multiprocessor Architecture for Mixed-Time-Criticality Applications*, pages 491–530. Springer Netherlands, Dordrecht, 2017
- Rasool Tavakoli and Majid Nabi. TIGeR: A Traffic-Aware Intersection-Based Geographical Routing Protocol for Urban VANETs. In *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, pages 1–5, June 2013